Fundamenta Informaticae 89 (2008) 479–510
IOS Press

On the Use of Behavioural Equivalences for Web Services' Development*

Filippo Bonchi, Antonio Brogi, Sara Corfini, Fabio Gadducci
Department of Computer Science
University of Pisa, Italy
{*fibonchi,brogi,corfini,gadducci*}*@di.unipi.it*

Abstract. Web services are emerging as a promising technology for the development of next generation distributed heterogeneous software systems. We define a new behavioural equivalence for Web services, based on bisimilarity and inspired by recent advances in the theory of reactive systems. The equivalence is compositional and decidable, and it provides a firm ground for enhanced behaviour-aware discovery and for a sound incremental development of services and service compositions.

1. Introduction

Web services are emerging as a promising technology for the development of next generation distributed heterogeneous software systems [33]. Roughly, a Web service is any piece of software that makes itself available over the Internet. A Web service is identified by a URI, it is universally accessible by means of standard protocols (WSDL [45], UDDI [42], SOAP [44]), and it self-describes its functionalities by exposing a public interface.

WSDL [45] is the currently employed standard for describing services. A WSDL description details what a service provides, by listing its operations in terms of input and output messages, and how a service can be invoked, by specifying one or more network locations where it can be accessed. WSDL descriptions do not include any information on the interaction *behaviour* of services, that is, on the order with which messages can be received or sent by each service. Unfortunately, the lack of behavioural information inhibits the possibility of *a priori* determining whether two services have the same behaviour as well as the possibility of verifying properties of service compositions, such as deadlock-freedom.

Various proposals have been recently put forward to feature more expressive service descriptions that include both semantics (viz., ontology-based) and behaviour information about services. One of the

*Partly supported by the EU FP6-IST IP 16004 SENSORIA and STREP 033563 SMEPP, and the MIUR FIRB TOCAI.IT.

major efforts in this direction is OWL-S [35], a high-level ontology-based language for describing services. Since OWL-S is a computer-interpretable semantic mark-up language providing all the needed information for describing services, OWL-S paves the way for the full automation of service discovery, invocation and composition. In particular, OWL-S service descriptions include a declaration of the interaction behaviour of services (the so-called *process model*), which provides the needed information for the *a priori* analysis and verification of service invocations and compositions.

The objective of this paper is to define a notion of *behavioural equivalence* for Web services. An immediate application of such a notion is the possibility of establishing whether syntactically different services feature the same behaviour, and hence for instance of verifying whether the upgrade of a service S with a new version S' may affect the interoperability with existing clients of S. The availability of a well-founded notion of behavioural equivalence can also be exploited to develop enhanced service discovery techniques so as to go beyond functional matching and determine whether a given service (or service composition) features a desired interaction behaviour. Two fundamental properties of any equivalence relation are *computability* and *compositionality*. Computability is a key requirement in ensuring the viability of an equivalence relation, that is, in allowing the development of automated software capable of determining whether two services are behaviourally equivalent or not. Compositionality permits to exploit the equivalence relation for a disciplined incremental development of services, by means of sound compositions and replacements. Namely, compositionality permits for instance to verify whether a sub-service P of a service S can be replaced by a new version P' without altering the behaviour of the entire service.

In this paper we first show how the behaviour of a Web service can be suitably described by means of a *Petri net*. Petri nets [39] are one of the best known and most widely adopted formalisms to express the concurrent behaviour of (software) systems. Besides providing a clear and precise semantics, Petri nets feature an intuitive graphical notation, and a number of techniques and tools for their analysis, simulation and execution are available. Petri nets have also been already employed to model Web services (e.g., see [17, 25, 3]). We introduce a simple variant of standard condition/event Petri nets (viz., CPR nets for Consume-Produce-Read nets) to naturally model the behaviour of Web services, and in particular the persistence of data. We then show how OWL-S process models can be directly mapped into CPR nets, borrowing from the translation from OWL-S to place/transition nets (P/T for short) described in [13].

Our next step is the identification of a suitable behavioural equivalence for our class of nets. Indeed, the dynamics of a Petri net, as well as those of most functional and process calculi, is usually defined in terms of reduction relations among its markings (viz., the states of a system). Despite its simplicity, the main drawback of reduction semantics is poor compositionality, in the sense that the dynamic behaviour of an arbitrary stand alone net may become unpredictable whenever it becomes a part of a larger net. Recently, Leifer and Milner [21] devised a methodology for distilling from a reduction relation a set of labels satisfying suitable requirements of minimality, in order to build a *labelled* reduction relation, such that the associated behavioural notion of *bisimilarity* is a compositional equivalence. The methodology has been later applied to P/T nets [30] as well as to their condition/event variant [41] (C/E for short). Thus, after discussing a motivating example, we define a novel notion of net equivalence, based on bisimilarity and inspired by the recent theoretical advances we just mentioned above. We show that this new equivalence relation is indeed compositional (i.e., it is a *congruence*), and that it is also decidable.

This paper is an extended version of [8]. Besides adding the formal proofs of the statements occurring there, the paper presents (1) the formal encoding from OWL-S to open CPR nets, only sketched in the original paper, as well as (2) a sample scenario discussing the issue of service replaceability, inspired by a

case study proposed by the European project SENSORIA. Moreover, (3) the streamlining of the decidable equivalence presented in [8], by proving that it coincides with weak bisimilarity: the latter equivalence is well-known in the literature, and thus easier to be understood and automatically checked with available tools. We believe that such a flexible notion of service equivalence allows for fruitful applications at the design level. It is worth mentioning here that in [9], exploiting the behavioural congruence introduced in [8] and making use of the formal encoding of OWL-S presented here, we extend the sample scenario (2) to outline a formal methodology addressing two main issues in Service-oriented Computing, i.e., publication of correct service specifications and replaceability of (sub)services.

The rest of the paper is organised as follows. In Section 2 we provide a short introduction to OWL-S, a formal definition of CPR nets and a first intuitive, yet informal encoding from OWL-S to CPR nets. The need of a weak, compositional behavioural equivalence for services is argued in Section 3, where some motivating examples are briefly discussed. Section 4 is devoted to present our notion of behavioural congruence, and to prove that it is both compositional and computable. Next, in Section 5, we define the formal encoding from OWL-S to open CPR nets. A sample scenario is presented in Section 6, where we employ our notion of service equivalence to tackle the issue of service replaceability. Related work is discussed in Section 7, while some concluding remarks are drawn in Section 8. Furthermore, we include the proof of the main theorem of the paper in Appendix A, and we discuss semi-saturated bisimulation (the computable service equivalence originally introduced in [8]) in Appendix B.

2. Modelling Web Services with Petri nets

Before showing our use of Petri nets for modeling Web services, we recall the essence of OWL-S, a high-level ontology-based language for describing Web services. An OWL-S service advertisement consists of three documents: the *service profile*, providing a high-level specification of the service by describing its functional (inputs and outputs) and extra-functional attributes; the *process model*, describing the service behaviour by providing a view of the service in terms of process composition; and the *service grounding*, stating how to interact with the service by specifying protocol and message format information.

In the paper we focus on the OWL-S process model, as it details the behavioural information needed to represent a service as a Petri net. More precisely, the process model describes a service as a composite process which consists, in turn, of composite processes and/or atomic processes. An atomic process cannot be decomposed further and it has associated inputs and outputs, while a composite process is built up by using a few control constructs: `sequence` (i.e., sequential execution), `choice` (conditional execution), `split` (parallel execution), `split+join` (parallel execution with synchronization), `any-order` (unordered sequential execution), `repeat-while` and `repeat-until` (iterative execution).

In [13] the second and third author proposed a mapping from OWL-S service descriptions to P/T nets with the objective to exploit the Petri net representation of services for a behaviour-aware service discovery. Given a query specifying the inputs and outputs of the service to be found, first the functional attributes of the available services are considered, in order to discover a composition capable of satisfying the query functionally. Next, the found composition is translated into a P/T net and analysed in order to verify properties such as deadlock-freedom.

The mapping presented in [13] takes into account the fact that an OWL-S atomic operation can be executed only if all its inputs are available and all the operations that must occur before its execution have been completed. Hence, atomic operations are mapped into transitions, and places and transition

firing rules are employed to model both the availability of data (i.e., the *data flow*) and the executability of atomic operations (i.e., the *control flow*). Indeed, an atomic operation T is modelled as a transition t having an input/output *data place* for each input/output of T, an *input control place* to denote that t is executable, as well as an *output control place* to denote that t has completed its execution.

However, the specific features concerning the Web service framework (already emphasised in [13]) suggest more specific nets. In this paper we introduce a simple variant of standard Petri nets [39] as a tool for properly modelling Web services. Given a net N modelling a Web service, we first of all noted that the portion of N restricted to transitions and control places should behave as a classical C/E net, that is, at most one token should occur for each place. Furthermore, we opted for a model stressing the persistence of data, meaning that, once a data has been produced by some service operation, it has to be kept available for all the service operations that input it. In other words, whilst control places can be produced and consumed, data places can be read, produced but not consumed. Hence, the behaviour of the portion of N restricted to data places recalls *contextual* nets [31].

To encompass into the same structure the different net behaviour determined by data and control places, we introduce our own particular flavour of contextual C/E nets, which is going to be formally presented in the following subsection.

We are going to discuss later a few alternative proposals for observational equivalence of Web services, often specified using BPEL, encoded into suitable Petri nets. It is worth mentioning here that a Petri nets semantics for DAML-S (the predecessor of OWL-S) was originally defined by Narayanan and McIlraith in [32]. The proposed mapping was incomplete, since the encoding of the `any-order` construct was not provided; most importantly, though, it exploited P/T nets, whose behaviour is quite different from our C/E based proposal. It is also worth mentioning the *E*-service nets [28], namely, nets equipped – similarly to our nets – with control places and input/output message places. Additional orchestration places are employed to compose different *E*-service nets. However, as for [32], also the proposal in [28] is not suitable to represent data persistency and, moreover, neither of them provide any (formal) definition of net composition.

2.1. Consume-Produce-Read nets

This subsection introduces *consume-produce-read* nets. These are a slight extension of standard Petri nets, since they are equipped with two disjoint sets of places, namely, the *control* (for consume-produce) places and the *data* (for produce-read) places.

Definition 2.1. (CPR net)
A *consume-produce-read* net (simply, *CPR net*) N is a five-tuple $(CP_N, DP_N, T_N, CF_N, DF_N)$ where

- CP_N is a finite set of *control places*,
- DP_N is a finite set of *data places* (disjoint from CP_N),
- T_N is a finite set of *transitions*,
- $CF_N \subseteq (CP_N \times T_N) \cup (T_N \times CP_N)$ is the *control flow relation*,
- $DF_N \subseteq (DP_N \times T_N) \cup (T_N \times DP_N)$ is the *data flow relation*.

Our nets behave as standard C/E nets with respect to control places, while, as we are going to see, data places are never emptied, once they are inhabited.

As for standard nets, we associate a *pre-set* and a *post-set* with each transition t, together with two additional sets, called *read-* and *produce-set*.

Definition 2.2. (pre-, post-, read-, and produce-set)
Given a CPR net N, we define for each $t \in T_N$ the sets

- ${}^{\diamond}t = \{s \in CP_N \mid (s,t) \in CF_N\}$
- $t^{\diamond} = \{s \in CP_N \mid (t,s) \in CF_N\}$
- ${}^{\bullet}t = \{s \in DP_N \mid (s,t) \in DF_N\}$
- $t^{\bullet} = \{s \in DP_N \mid (t,s) \in DF_N\}$

which denote respectively the *pre-set*, *post-set*, *read-set* and *produce-set* of t.

Figure 1 depicts our graphical notation. Diamonds, circles and rectangles represent control places, data places and transitions, respectively. The transition in Figure 1 *reads* the data places labelled $I_1, I_2, \ldots, I_n$ (shown by straight lines) and *produces* the data places labelled $O_1, \ldots, O_m$ (shown by pointed arrows). In doing so, the control flow passes from the left-most to the right-most control place.

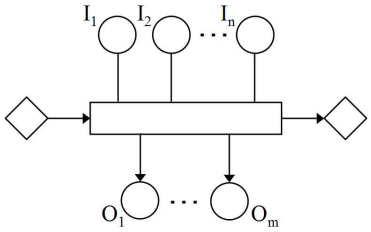

Figure 1. Modelling atomic operations as CPR net transitions.

Definition 2.3. (marking)
Given a CPR net N, a *marking* M for N is a finite subset of $P_N = CP_N \cup DP_N$.

A marking of the net N coincides with a subset of its set P_N of *places*, since each place can contain at most one token. The evolution of a net is given by a relation over markings. A transitions t is *enabled* by a marking M if the control places which belong to the pre-set of t as well as the data places which belong to the read-set of t are contained in M, and no overlap (as defined later) between M and the post-set of t occurs. In this case a *firing step* may take place: t removes the tokens from the control places which belong to the pre-set of t and adds a token to each place which belongs to the post- and produce-set of t.

Definition 2.4. (firing step)
Let N be a CPR net. Given a transition $t \in T_N$ and a marking M for N, a *firing step* is a triple $M[t\rangle M'$ such that $({}^{\diamond}t \cup {}^{\bullet}t) \subseteq M$ and $(M \cap t^{\diamond}) \subseteq {}^{\diamond}t$ (M *enables* t), and moreover $M' = (M \setminus {}^{\diamond}t) \cup t^{\diamond} \cup t^{\bullet}$.

We write $M[\rangle M'$ if there exists some t such that $M[t\rangle M'$.

The enabling condition states that (i) all the tokens of the pre-set of a transition have to be contained in the marking, and (ii) that the marking does not contain any token in the post-set of the transition, unless it is consumed and regenerated. The second condition characterizes C/E nets. Data places act instead as sinks, hence, any token can be added and only the occurrence of a token is checked. The read-only feature of data places is reminiscent of so-called *contextual* C/E nets by Montanari and Rossi [31].

2.2. From OWL-S to CPR nets

After introducing CPR nets, we informally discuss how OWL-S service descriptions can be mapped into CPR nets. For the sake of clarity, a formal presentation of the encoding is delayed until Section 5, after the introduction of *open* CPR nets and of their semantics.

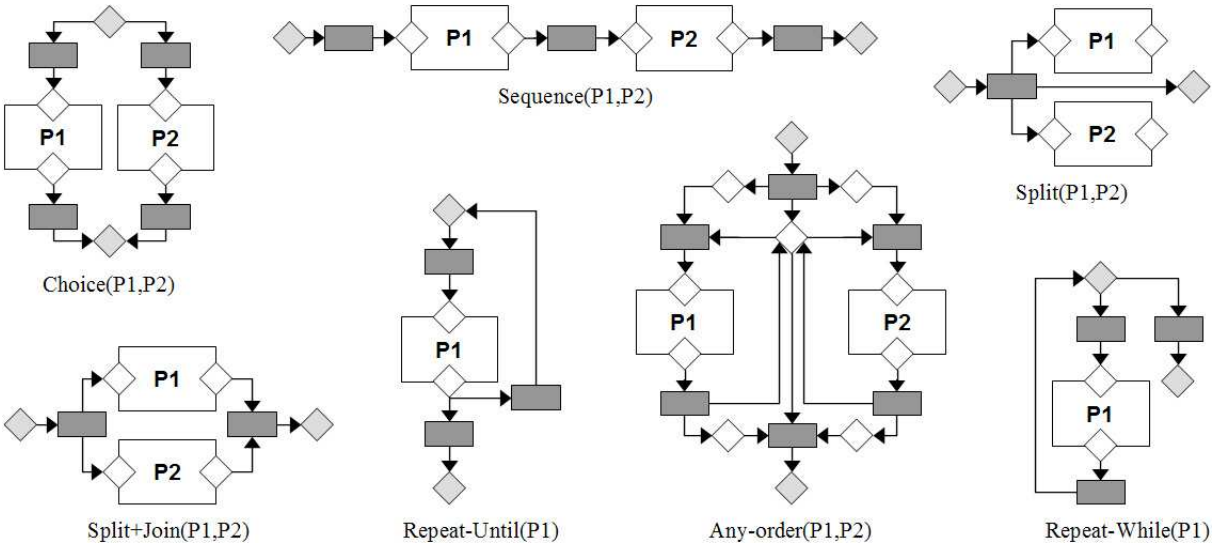

Figure 2. Modelling OWL-S composite operations as CPR nets.

Let us define a service as a triple (i, P, f) where P denotes the CPR (sub)net representing the service and i and f denote the initial and the final control places of P, respectively. To define compositional operators it is sufficient to properly coordinate the initial and final control places of the employed services. For instance, let us consider the *sequential composition* of two services $(i_1, P1, f_1)$ and $(i_2, P2, f_2)$. This is a CPR net consisting of the two services and a transition whose starting control place is f_1 and the final control place is i_2. By doing so, $P1$ has to be completed before $P2$ can start.

The reader can intuitively understand how OWL-S composite operations can be mapped into CPR nets by observing Figure 2 (some of them are suggested by [4]). The PX-labelled boxes represent (i_x, PX, f_x) services, the dark gray rectangles identify empty transitions, and the light gray diamonds denote the starting and final control places of the resulting nets. To simplify reading we omitted data places from the nets of Figure 2. Yet, it is important to note that the PX-boxes can share data places to simulate the exchange of data amongst services, as formally defined in Section 4.1.

3. Motivating examples

The ultimate objective of this paper is to introduce a decidable and compositional notion of equivalence between Web services represented as CPR nets. This notion establishes whether two service behaviours are equivalent and it can be employed to suitably address the following issues:

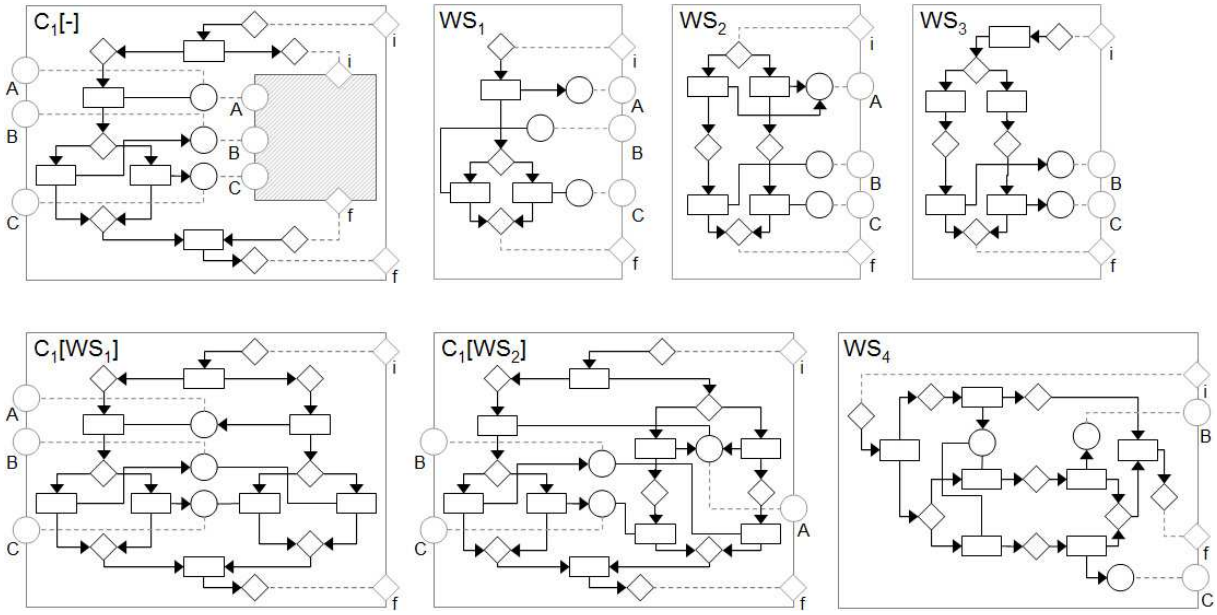

Figure 3. Example of (non-)equivalent services.

1) *incremental development of services* — to check whether two different versions of a service are equivalent;

2) *publication of correct service specifications* — to check whether a (complex) service implements a given specification;

3) *matching services* — to check whether a (composition of) service(s) matches a query that specifies the behaviour of the desired service (composition) to be found (e.g., the notion of equivalence needed by [13]);

4) *replaceability of services* — to check whether a service s which takes part in a composition $C[s]$ can be replaced with a different service r without changing the behaviour of C, i.e., guaranteeing the equivalence between $C[s]$ and $C[r]$.

We can hence outline the main features that a suitable notion of equivalence should have, that is, *weakness* and *compositionality*. It has to be weak as it must equate services with respect to their externally observable behaviour. Indeed, it is reasonable that two versions of a service differently implement the same operations (1), that a service specification hides unnecessary and/or confidential details of its implementation (2), as well as that a simple query can be satisfied by a complex service composition (3).

Therefore, this notion of equivalence has to be abstract enough to equate services that differ only on internal transition steps. Furthermore, (4) also asks for compositionality, and if two services are equivalent, then they can be always used interchangeably.

Let us now consider some examples (inspired by [26]). Figure 3 illustrates the CPR nets of seven services, where rectangles, circles and diamonds represent transitions, data places and control places, respectively. Note the boxes that limit each service. As we formalise in the following section, a box represents the *outer* interface of a service, that is, the set of places which can interact with the environment. Hence, those places that in Figure 3 lie on the border of boxes are the ones that can be observed externally. Consider the services WS_1 and WS_2 of Figure 3. As one may note, WS_1 and WS_2 have the same behaviour with respect to the notion of *trace equivalence*. Indeed, they have identical sets of traces, since after producing A they may alternatively read either B or C. Consider now the context $C_1[-]$, depicted in Figure 3, which represents a possible environment in which WS_1 and WS_2 can be embedded. Note the gray area contained in $C_1[-]$. Its border is the *inner* interface of $C_1[-]$. As formalised in Section 4.1, a service WS can be inserted inside a context $C[-]$ if the outer interface of WS and the inner interface of $C[-]$ coincide. The resulting composition $C[WS]$ consists of the fusion of such interfaces, as well as of the fusion of the data places and the union of the transitions of WS and $C[-]$. We note that $C_1[-]$ inputs A, produced by WS_1, and yields B or C, taken as input by WS_1. Hence, the composition $C_1[WS_1]$ works and finishes properly. Now, in order to test if the trace equivalence is the notion suitable for our purpose, we replace WS_1 with the trace equivalent service WS_2 and we check whether the composition $C_1[WS_2]$ works properly as well. Yet, $C_1[WS_2]$ produces a possibly deadlocking system.

Let us now describe a second example arguing for the need of weakness. Consider the services WS_3 and WS_4. For instance, WS_4 could be a composition candidate to satisfy the query represented by WS_3. Although WS_3 and WS_4 appear different as they perform a different number of transitions, they both produce B or C. Namely, WS_3 and WS_4 have identical externally observable behaviour, and they indeed should be considered equivalent.

By taking into account the requirements briefly outlined in this motivating section, we define a novel notion of equivalence based on bisimilarity which features weakness, compositionality and decidability.

4. A congruence for Open CPR nets

The previous section argued about the relevance of a behavioural equivalence, formally characterizing the notion of replaceability, and motivated why such an equivalence should be compositional and weak.

A first step for defining compositionality is to equip nets with a notion of interface and context. Next, we introduce two notions of equivalence: the first is conceptually the correct one, even if it turns out to be quite hard to reason about, while the second provides a simple, decidable characterization of the former.

4.1. Open CPR nets and CPR contexts

For the sake of presentation, a chosen net $N = (CP_N, DP_N, T_N, CF_N, DF_N)$ is assumed.

Definition 4.1. (Open CPR net)
Let N be a CPR net. An *interface* for N is a triple $\langle i, f, OD \rangle$ such that $i \neq f$ and

- i is a control place (i.e., $i \in CP_N$), the *initial* place;

- f is a control place (i.e., $f \in CP_N$), the *final* place; and
- OD is a set of data places (i.e, $OD \subseteq DP_N$), the *open* data places.

An interface is an *outer interface* O for N if there exists no transition $t \in T_N$ such that either $i \in t^\diamond$ or $f \in {}^\diamond t$. An *open* CPR net $\mathcal{N}$ is a pair $\langle N, O \rangle$, for N a CPR net and O an outer interface for N.

Next, we symmetrically define an *inner interface* for N as an interface such that there is no transition $t \in T_N$ verifying either $f \in t^\diamond$ or $i \in {}^\diamond t$.

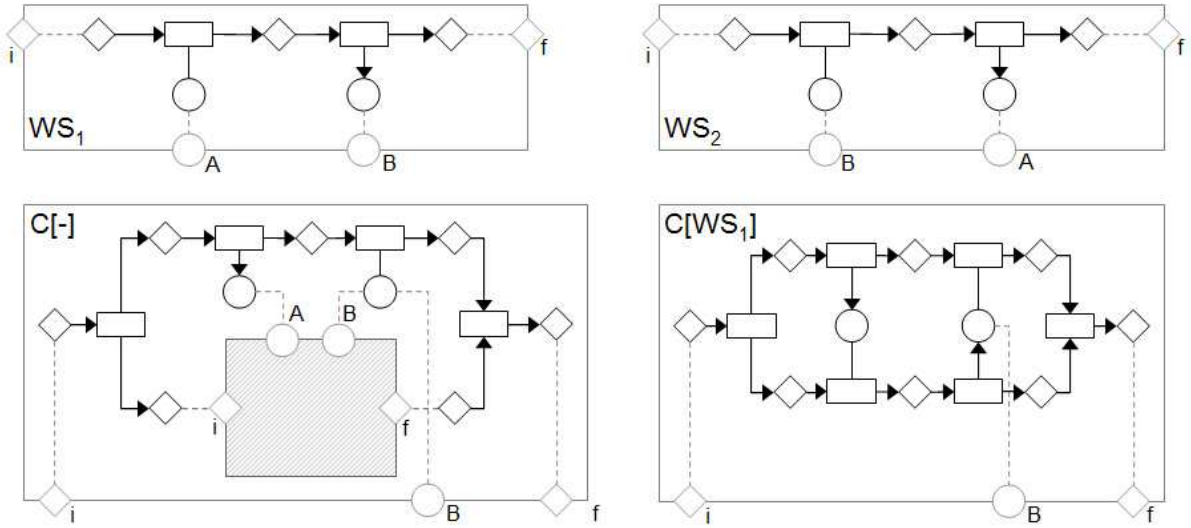

Figure 4. Two open nets, a context and a composite net.

The condition characterizing outer interfaces requires that the initial place has no incoming transition and the final place has no outgoing transition. Hence, Open CPR (OCPR for short) nets recall the WorkFlow (WF) nets advocated by van der Aalst [43][1] for modeling Web services.

The components of a specific interface or open net are often denoted by adding them a subscript. Given an open net $\mathcal{N}$, $Op(\mathcal{N})$ denotes the set of *open places*, which consists of those places occurring on the interface, initial and final places included. Furthermore, the places of $\mathcal{N}$ not belonging to $Op(\mathcal{N})$ constitute the *closed places*. It is important to note that open places are crucial for the definition of the observational equivalence that we propose in the next subsection.

Figure 4 shows the graphical notation used to represent OCPR nets. The bounding box of the OCPR net WS_1 represents the outer interface of the net. Note the initial and final places used to compose the control of services (as suggested in Figure 2) as well as the open data places employed to share data.

Definition 4.2. (CPR context)
A CPR *context* $C[-]$ is a triple $\langle N, O, I \rangle$ such that N is a CPR net, $I = \langle i_I, f_I, OD_I \rangle$ and $O = \langle i_O, f_O, OD_O \rangle$ are an inner and an outer interface for N, respectively, and $i_I \neq f_O$, $i_O \neq f_I$.

[1] More precisely, our nets lack the connectiveness requirement, see e.g. [43, Sect. 2.2]. Note however that, even if this is not made explicit, the property holds for all the nets obtained by modeling OWL-S composite operations.

A context $C[-]$ is shown in Figure 4. Basically, it is an open net with a hole, represented there by a gray area. The border of the hole denotes the inner interface of the context. As for open nets, the bounding box is the outer interface. The only difference between inner and outer interfaces is that the initial place of the former has no outgoing transitions (and vice versa for final places).

Contexts represent environments into which services can be embedded, i.e., possible ways they can be used by other services. In order for an OCPR net to be inserted in a context, the net outer interface and the context inner interface must coincide.

Definition 4.3. (CPR composition)
Let $\mathcal{N} = \langle N, O\rangle$ be an OCPR net and $C[-] = \langle N_C, O_C, I_C\rangle$ a CPR context, such that $O = I_C$. Then, the *composite net* $C[\mathcal{N}]$ is the OCPR net $(CP_N \uplus_O CP_{N_C}, DP_N \uplus_O DP_{N_C}, T_N \uplus T_{N_C}, CF_N \uplus CF_{N_C}, DF_N \uplus DF_{N_C})$ with outer interface O_C.

The disjoint union of the two nets is performed, except for the places occurring in O, which are coalesced (hence the symbol $\uplus_O$). Moreover, O_C becomes the set of open places of the resulting net.

Consider for instance the net WS_1, the context $C[-]$ and their composition, denoted by $C[WS_1]$, as illustrated in Figure 4. The places on the outer interface of $WS1$ are coalesced with the ones on the inner interface of $C[-]$. The outer interface of $C[WS_1]$ is the outer interface of $C[-]$. Note that the data place A is open in WS_1 and closed in $C[WS_1]$: this example highlights the capability of hiding places, removing them from the outer interface of an open net. Indeed, this feature is reminiscent of the restriction operator of process calculi, such as CCS [29].

It is worth observing that contexts are composable: CPR contexts form a category where interfaces are objects and contexts are arrows going from the inner interface to the outer interface[2] and OCPR nets are arrows whose source is the empty interface and target is the outer interface.

4.2. Saturated bisimilarity for OCPR nets

This section addresses the question of the equivalence between nets. Our answer relies on an observational approach, equating two systems if they can not be told apart by an external observer. More precisely, the observer can only examine the open places of a net, which is otherwise a black box, and only those places may be checked for verifying whether they are actually inhabited or if they are empty.

For the sake of presentation, a chosen OCPR net $\mathcal{N} = \langle N, O\rangle$ is assumed.

Definition 4.4. (observation)
Let $\mathcal{N}$ be an OCPR net, and M a marking of N. The *observation* on $\mathcal{N}$ at M is the set of places $Obs(\mathcal{N}, M) = Op(\mathcal{N}) \cap M$.

Thus, an observer can look at the evolution of the system by observing if tokens are either produced or consumed in the open places. Accordingly, two OCPR nets $\mathcal{N}$ and $\mathcal{N}'$ with the same outer interface and with initial markings M and M' are considered equivalent if $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$ and if every state reachable from M in N is equivalent to a state reachable from M' in N' (and vice versa).

The previous remark is formalized below, where $\mathcal{MN}$ denotes the set of all OCPR nets with markings and $\twoheadrightarrow_{\mathcal{N}}$ denotes the reflexive and transitive closure of the firing relation $[\rangle$ for the net N underlying $\mathcal{N}$.

[2]CPR nets and their morphisms (not defined here) form an *adhesive category* [19]. The category of CPR contexts is the *cospan bicategory* over such category [40], thus it is amenable to the borrowed context technique [16] for distilling a set of labels from a reduction relation. That technique is implicitly exploited in the next section.

Definition 4.5. (naive bisimulation)
A symmetric relation $\mathfrak{R} \subseteq \mathcal{MN} \times \mathcal{MN}$ is a *naive bisimulation* if whenever $(\mathcal{N}, M)\ \mathfrak{R}\ (\mathcal{N}', M')$ then

- $O_{\mathcal{N}} = O_{\mathcal{N}'}$ and $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$, and
- $M \twoheadrightarrow_{\mathcal{N}} M_1$ implies $M' \twoheadrightarrow_{\mathcal{N}'} M_1'$ & $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$.

The union of all naive bisimulations is called *naive bisimilarity*.

The equivalence above is "naive" in the sense that it clearly fails to be compositional. Indeed, consider the OCPR nets WS_1 and WS_2 in Figure 4. They are trivially equivalent since none of them can fire. However, if we insert them into a context with a transition generating a token in the initial place and in the data place A, we obtain two different contexts (one can now produce a token on B reaching the final state f, while the latter can not move).

The solution out of the impasse, which is quite standard in functional languages and process calculi, is to allow the observer to perform more complex experiments, inserting a net into any possible context.

Definition 4.6. (saturated bisimulation)
A symmetric relation $\mathfrak{R} \subseteq \mathcal{MN} \times \mathcal{MN}$ is a *saturated bisimulation* if whenever $(\mathcal{N}, M)\ \mathfrak{R}\ (\mathcal{N}', M')$ then

- $O_{\mathcal{N}} = O_{\mathcal{N}'}$ and $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$, and
- $\forall C[-].M \twoheadrightarrow_{C[\mathcal{N}]} M_1$ implies $M' \twoheadrightarrow_{C[\mathcal{N}']} M_1'$ & $(C[\mathcal{N}], M_1)\ \mathfrak{R}\ (C[\mathcal{N}'], M_1')$.

The union of all saturated bisimulations is called *saturated bisimilarity* ($\approx_S$).

Clearly, $\approx_S$ is by definition a congruence. Indeed, it is the largest bisimulation that preserves compositionality, as stated below.

Proposition 4.1. $\approx_S$ is the largest bisimulation that is also a congruence.

The above proposition ensures the compositionality of the equivalence, hence, the possibility of replacing one service by an equivalent one without changing the behaviour of the whole composite service. Moreover, the equivalence is "weak" in the sense that, differently from most of the current proposals, no explicit occurrence of a transition is observed. The previous definition leads to the following notion of equivalence between OCPR nets, hence, between services.

Definition 4.7. (bisimilar nets)
Let $\mathcal{N}, \mathcal{N}'$ be OCPR nets. They are *bisimilar*, denoted by $\mathcal{N} \approx \mathcal{N}'$, if $(\mathcal{N}, \emptyset) \approx_S (\mathcal{N}', \emptyset)$.

The choice of the empty marking guarantees that the equivalence is as general as possible. Indeed, the presence of a token in an open place can be simulated by closing the net with respect to a transition adding a token in that place, and if any two nets are saturated bisimilar with respect to the empty marking, they are so also with respect to any marking with tokens in the open places.

The negative side of $\approx$ is that this equivalence seems quite hard to be automatically decided because of the quantification over all possible contexts. In the following subsection we introduce an alternative equivalence, easier to reason about and to automatically verify, and we prove that it coincides with $\approx_S$.

4.3. An equivalent decidable bisimilarity

Saturated bisimulation seems conceptually the right notion, and this is further argued in Section 7. However, it seems quite hard to analyze (or automatically verify), due to the universal quantification over contexts. In this section we thus introduce weak bisimilarity, based on a simple *labelled transition system* (LTS) distilled from the firing semantics of an OCPR net.

The introduction of a LTS is inspired to the theory of *reactive systems* [21]. This meta-theory suggests guidelines for deriving a LTS from an unlabelled one, choosing a set of labels with suitable requirements of minimality. In the setting of OCPR nets, the reduction relation is given by $[\rangle$, and a firing is allowed if all the preconditions of a transition are satisfied. Thus, intuitively, the minimal context that allows a firing just adds the tokens needed for that firing.

Definition 4.8. (labelled transition system)
Let $\mathcal{N}$ be an OCPR net and $\Lambda = \{\tau\} \cup (\{+\} \times P_N) \cup (\{-\} \times CP_N)$ a set of labels, ranged over by l. The *transition relation* for $\mathcal{N}$ is the relation $R_{\mathcal{N}}$ inductively generated by the set of inference rules below

$$\frac{o \in Op(\mathcal{N}) \setminus (M \cup \{f\})}{M \stackrel{+o}{\rightarrow}_{\mathcal{N}} M \cup \{o\}} \qquad \frac{f \in M}{M \stackrel{-f}{\rightarrow}_{\mathcal{N}} M \setminus \{f\}} \qquad \frac{M [\rangle M'}{M \stackrel{\tau}{\rightarrow}_{\mathcal{N}} M'}$$

where $M \stackrel{l}{\rightarrow}_{\mathcal{N}} M'$ means that $\langle M, l, M' \rangle \in R_{\mathcal{N}}$, and i, f denote the initial and the final place of $\mathcal{N}$, respectively.

Thus, a context may add tokens in open places, as represented by the transition $\stackrel{+o}{\rightarrow}_{\mathcal{N}}$, in order to perform a firing. Similarly, a context may consume tokens from the final place f. A context cannot interact with the net in other ways, since the initial place i can be used by the context only as a post condition and the other open places are data places whose tokens can be read but not consumed. Instead, τ transitions represent internal firing steps, i.e., steps needing no additional token from the environment.

The theory of reactive systems ensures that, for a suitable choice of labels, the (strong) bisimilarity on the derived LTS is a congruence [21]. However, often that bisimilarity does not coincide with the saturated one. In the case at hand, we introduce a notion of weak bisimilarity, abstracting away from the number of steps performed by nets, that indeed coincides with the saturated one.

Definition 4.9. (weak bisimulation)
A symmetric relation $\mathfrak{R} \subseteq \mathcal{MN} \times \mathcal{MN}$ is a *weak bisimulation* if whenever $(\mathcal{N}, M)\ \mathfrak{R}\ (\mathcal{N}', M')$ then

- $O_{\mathcal{N}} = O_{\mathcal{N}'}$ and $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$,
- $M \stackrel{+o}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \stackrel{+o}{\rightarrow}_{\mathcal{N}'} M_1'$ & $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$,
- $M \stackrel{-f}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \stackrel{-f}{\rightarrow}_{\mathcal{N}'} M_1'$ & $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$, and
- $M \stackrel{\tau}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \twoheadrightarrow_{\mathcal{N}'} M_1'$ & $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$.

The union of all weak bisimulations is called *weak bisimilarity* ($\approx_W$).

The key theorem of the paper is stated below.

Theorem 4.1. $\approx_S = \approx_W$.

Thus, in order to prove that two OCPR nets are bisimilar, it suffices to exhibit a weak bisimulation between the states of the two nets that includes the pair of empty markings. Most importantly, though, this verification can be automatically performed, since the set of possible states of an OCPR net are finite. Hence, the result below immediately follows.

Corollary 4.1. $\approx_S$ is decidable.

5. A compositional encoding for OWL-S

This section presents the formal encoding of OWL-S service descriptions into OCPR nets: it first introduces the notion of binary context, and then use it for modelling composite services. This section is rounded up by a simple result on *hiding* that is useful for the sample scenario discussed in Section 6.

5.1. On binary contexts

As depicted in Figure 1, an atomic process is encoded by a net containing a single transition. Instead, the encoding of a composite service requires to extend the notion of interface, in order to accommodate the plugging of two nets into a context.

Definition 5.1. (binary contexts)
A CPR *binary* context $C[-_1, -_2]$ is a 4-tuple $\langle N, O, I_1, I_2\rangle$ such that the triples $\langle N, O, I_1\rangle$, $\langle N, O, I_2\rangle$ are CPR contexts, and $\{i_{I_1}, f_{I_1}\} \cap \{i_{I_2}, f_{I_2}\} = \emptyset$.

Since it suffices for our purposes, we restrict our attention to binary contexts, the general definition being easily retrieved. In addition, it is explicitly required that the control places of the inner interfaces are all different, while no condition is required for data places.

Definition 5.2. (binary composition)
Let $\mathcal{N}_1 = \langle N_1, O_1\rangle$, $\mathcal{N}_2 = \langle N_2, O_2\rangle$ be OCPR nets, and $C[-_1, -_2]$ a CPR binary context, such that $O_1 = I_1$ and $O_2 = I_2$. Then, the *composite net* $C[\mathcal{N}_1, \mathcal{N}_2]$ is the OCPR net $(CP_{N_1} \uplus_U CP_{N_2} \uplus_U CP_N, DP_{N_1} \uplus_U DP_{N_2} \uplus_U DP_N, T_{N_1} \uplus T_{N_2} \uplus T_N, CF_{N_1} \uplus CF_{N_2} \uplus CF_N, DF_{N_1} \uplus DF_{N_2} \uplus DF_N)$ with outer interface O_C.

As for unary contexts, the disjoint union of the three nets is performed, except for coalescing those places occurring either in O_1 or O_2 (denoted by $\uplus_U$).

5.2. Presenting the encoding

In order to define the encoding, we first introduce the notion of context extension.

Definition 5.3. (context extension)
Let A be a set of data places and let $C[-] = \langle N, O, I\rangle$ be a context. The context extension $C_A[-]$ is the context whose net is $N_A = (CP_N, DP_N \uplus_U A, T_N, CF_N, DF_N)$, inner interface is $I_A = \langle i_I, f_I, OD_I \cup A\rangle$ and outer interface is $O_A = \langle i_O, f_O, OD_O \cup A\rangle$.

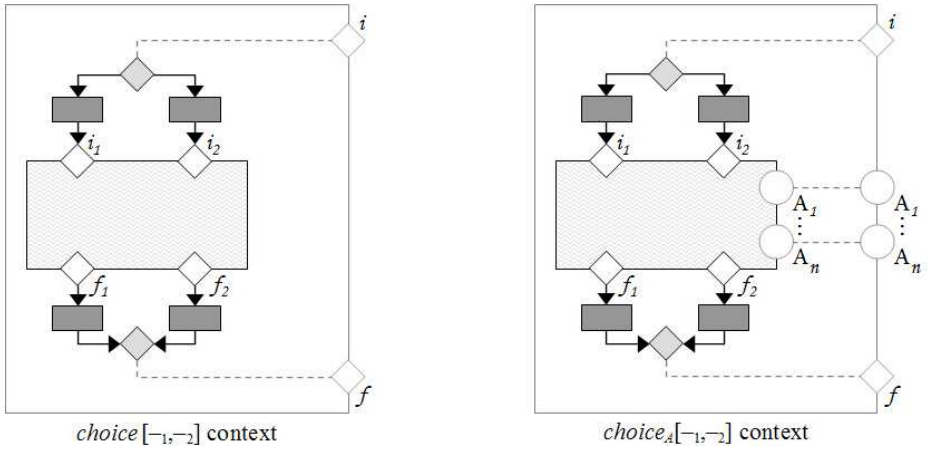

Figure 5. Binary contexts corresponding to $choice$ (left) and its extension for $A = \{A_1, \ldots A_n\}$ (right).

Thus, A is actually a set of data places, and the data places of the resulting context are obtained by disjoint union with respect to DP_N (and implicitly with respect to CP_N as well), except for coalescing those data places occurring in either in O_1 or O_2 (denoted by $\uplus_U$). An analogous operation is defined for binary contexts. As an example, consider the contexts $choice[-_1, -_2]$ and $choice_A[-_1, -_2]$ for $A = \{A_1, \ldots, A_n\}$ illustrated in Figure 5. The latter is the extension of the former with respect to set A.

In order to define the encoding, for each OWL-S operator `op` we need to define a corresponding (possibly binary) context $op[-]$. The contexts in Figure 6 depict the chosen encodings for all the OWL-S operators, with the exception of $choice$, that is instead presented in Figure 5. The meaning of such contexts is quite intuitive, and this mapping is clearly subsuming the informal encoding presented in Figure 2. Note also that the contexts actually depicted in Figure 6 are the extensions $op_A[-]$ of contexts $op[-]$ corresponding to OWL-S operators `op`.

Now we can give the formal encoding. Let S be an OWL-S process model and let A be a set of data places, containing all the data occurring in S. The encoding of S with respect to the set A of open places is inductively defined as follows

$$\|S\|_A = \begin{cases} \mathcal{N}_{S,A} & \text{if } S \text{ is atomic,} \\ op_A[\|S_1\|_A] & \text{if } S = \mathtt{op}(S_1), \\ op_A[\|S_1\|_A, \|S_2\|_A] & \text{if } S = \mathtt{op}(S_1, S_2), \end{cases}$$

where $\mathcal{N}_{S,A}$ is the OCPR net with a single transition that reads all the input data of the atomic service S, and produces all the output data of S (as illustrated by Figure 1), extended with all the data places of A.

For the sake of simplicity, we left implicit the possible renaming of control places, that may be needed for the composition of nets and contexts to be well-defined. Note that the translation can be made automatic: a prototypical tool[3], translating OWL-S service descriptions into OCPR nets (described by a XML file), has been recently presented in [14].

With respect to our case study, it is worth noting that the encoding of conditional execution, viz., `if-then-else`, coincides with the encoding of non-deterministic execution, viz., `choice`. Indeed, our implementation of the process model abstracts away from data values.

[3]Available at `http://www.di.unipi.it/~corfini/owls2pnml.html`.

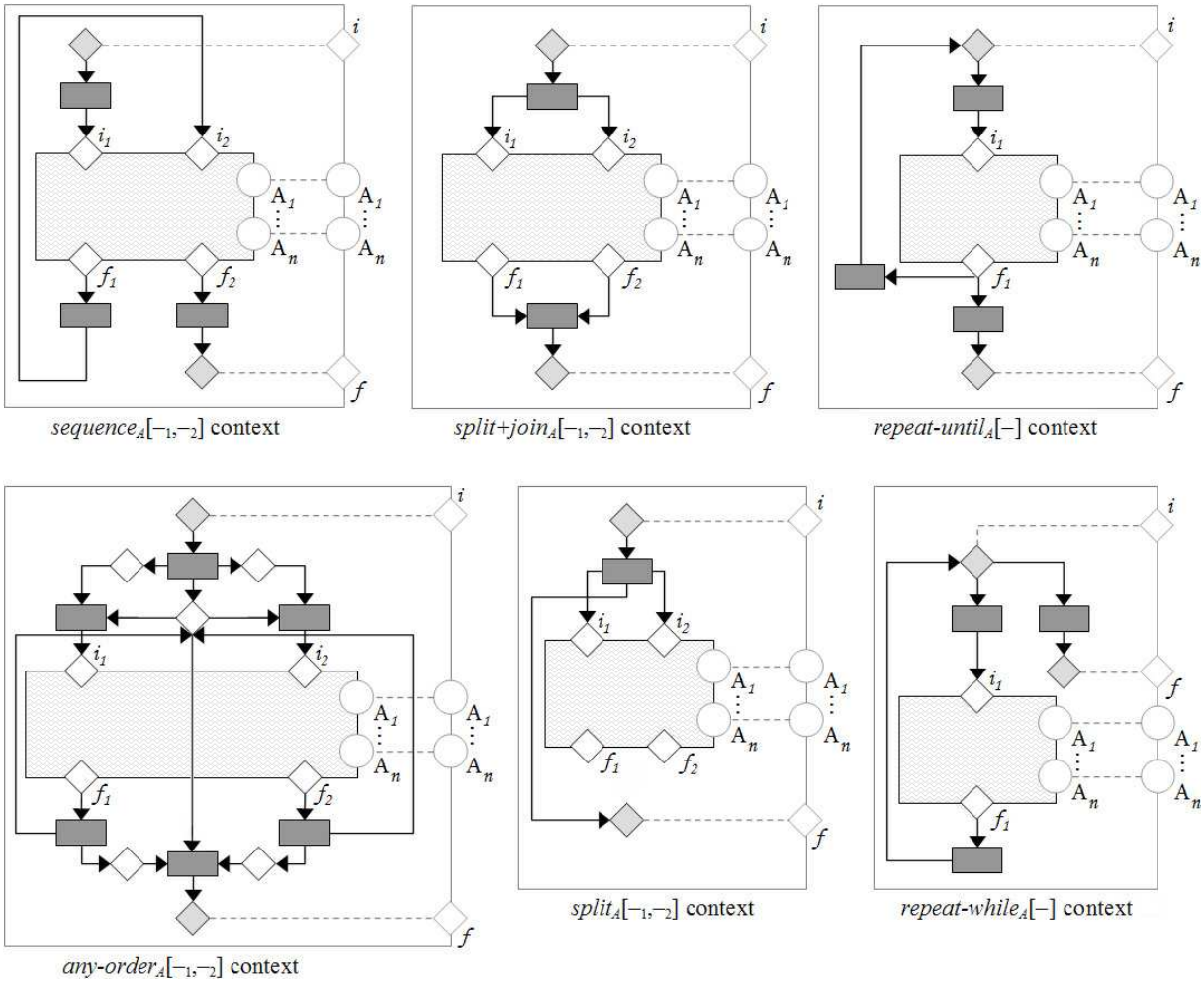

Figure 6. Contexts corresponding to OWL-S operators extended for $A = \{A_1, \ldots A_n\}$.

5.3. Hiding data places

The above presented encoding maps an OWL-S service into an OCPR net, where all the data places are open, i.e., they belong to the outer interface. Proposition 5.1 will turn out to be of use for those cases where it might be necessary to abstract away from irrelevant/confidential data.

Definition 5.4. (hiding operator)
Let $O = \langle i_O, f_O, OD_O \rangle$ be an outer interface and A a set of data places such that $A \subseteq OD_O$. The *hiding* operator (with respect to A and O) $\nu_{A,O}$ is the context whose net has no transition, inner interface is O and outer interface is $\langle i_O, f_O, OD_O \setminus A \rangle$.

For the sake of readability, we leave out the second index of a restriction operator, whenever it is clear from the context. As an example, consider the OCPR nets depicted in Figures 11 and 13. The

latter is obtained from the former by simply hiding the (sub)set of open data places $A = \{$firstRating, secondRating, thirdRating$\}$, i.e., by applying the hiding operator ν_A.

We round up the section with a simple result on disjoint compositionality, which is needed later on when discussing about (sub)service replaceability.

Proposition 5.1. Let $\mathcal{N}$ be a net, $A \subseteq Op(\mathcal{N})$ a set of data places, and $C[-] = \langle N, O, I \rangle$ a context such that $OD_O \cap A = \emptyset$. If $C[\nu_A[\mathcal{N}]]$ is well-defined, then $\nu_A[C_A[\mathcal{N}]] \approx_S C[\nu_A[\mathcal{N}]]$.

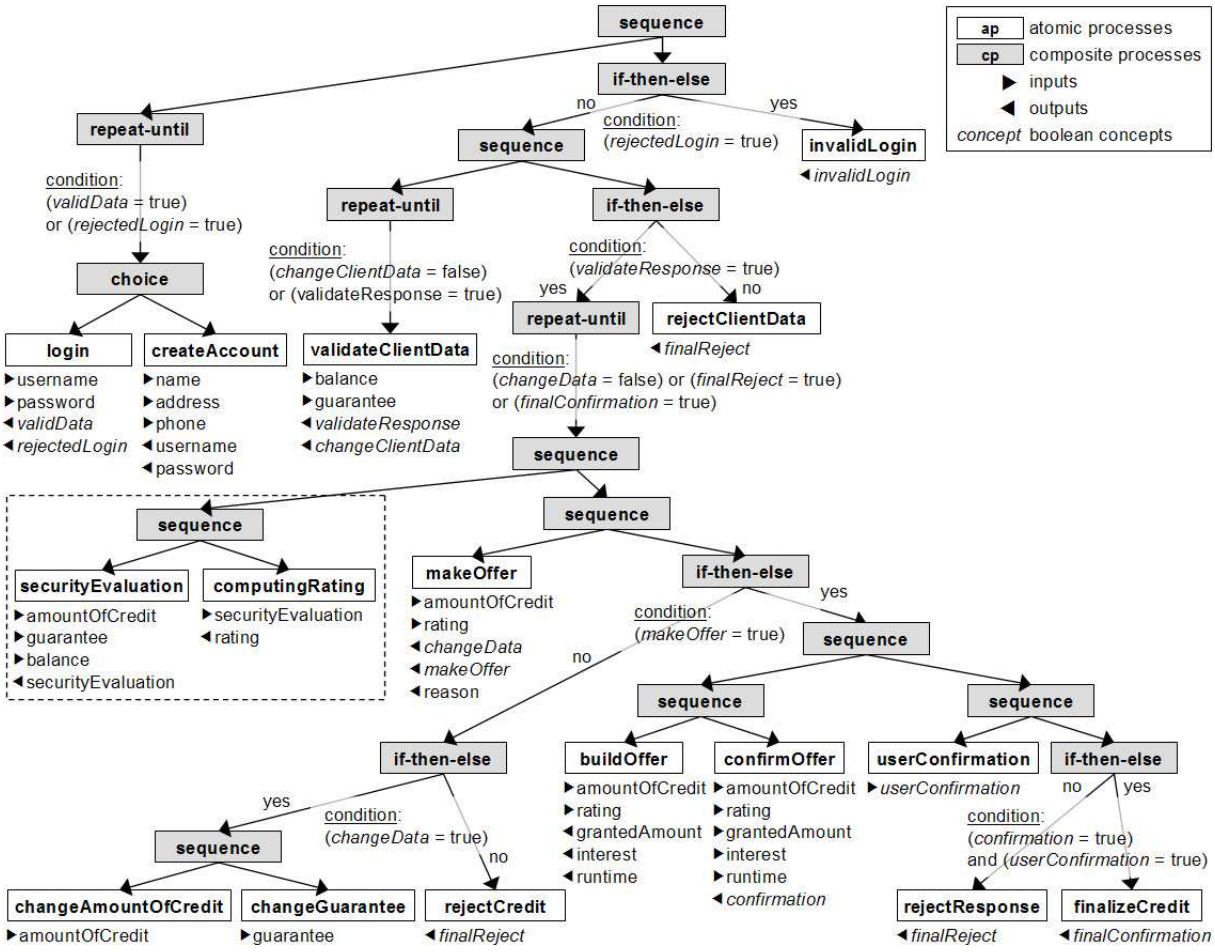

Figure 7. The OWL-S process model of the CreditPortal service.

In plain terms, removing the places in A from the interface of a net $\mathcal{N}$, and then inserting the resulting net in a context $C[-]$, has the same effect as inserting $\mathcal{N}$ in a slightly enlarged context $C_A[-]$, and later removing those same places.

6. A sample scenario on service replaceability

As we anticipated in Section 3, the notion of service equivalence introduced in this paper can be suitably employed to address some crucial issues such as (1) incremental development of services, (2) publication of correct service specifications, (3) matching services and (4) replaceability of services. In particular, in this section, we illustrate how the theoretical results presented in Section 4 can be employed to resolve the issue of service replaceability. To this end, a concrete sample scenario[4] concerning banking systems is introduced.

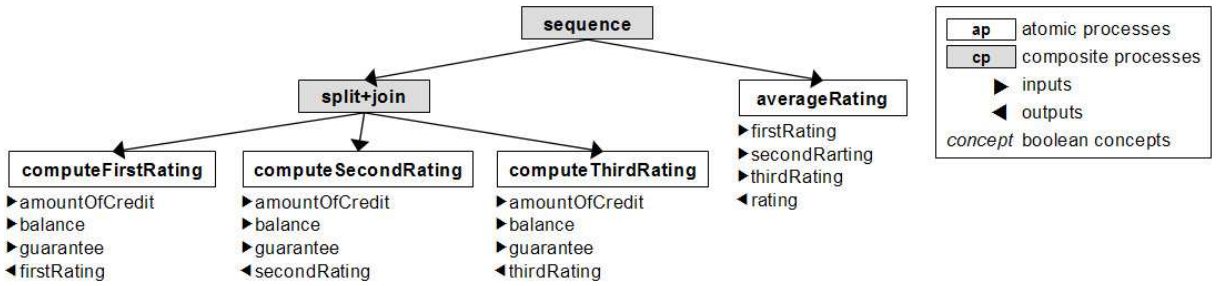

Figure 8. The OWL-S process model of the RatingOne service.

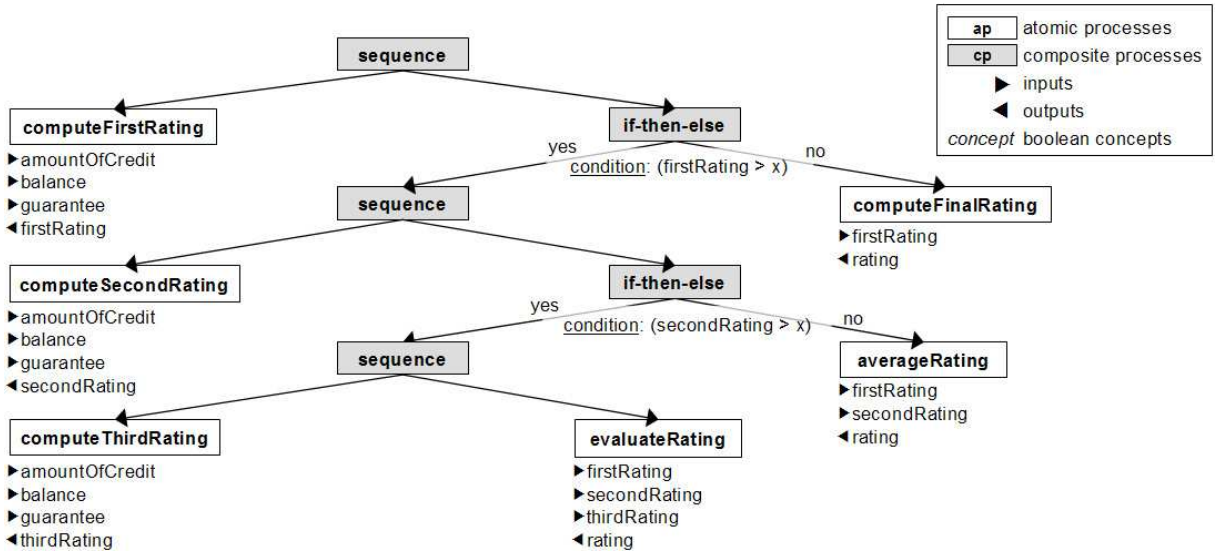

Figure 9. The OWL-S process model of the RatingTwo service.

Let us first consider the CreditPortal service, that grants loans to bank customers. CreditPortal implements three steps, namely, (1) authentication of the customer and upload of her/his personal data,

[4]The banking scenario – whose extended description can be found in [9] – is inspired by the finance case study by S&N AG netBank solutions (`http://www.s-und-n.de/`), one of the enterprises involved in the SENSORIA project (`http://www.sensoria-ist.eu/`).

(2) evaluation of the customer credit request, and (3) formulation of the loan offer. In the first step, after logging into the bank system, the customer has to upload information regarding balance and offered guarantees. In the second step, CreditPortal evaluates the customer reliability and computes a rating of the credit request. Finally, in the last step CreditPortal either decides to grant the loan to the customer and to build an offer, or it rejects the credit request.

The OWL-S specification of CreditPortal is available at [1]. Yet, for the convenience of the reader, we show in Figure 7 a more compact tree representation of the CreditPortal process model. Each internal node is labelled with the type of the composite process it represents, and, in case of conditional and iterative control constructs, also with a condition. Each leaf is associated with the inputs and the outputs of the corresponding atomic process. It is worth noting that in the actual OWL-S description of CreditPortal each input and output parameter is annotated with a concept defined in a shared ontology.

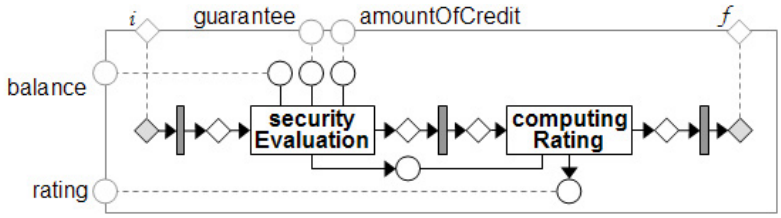

Figure 10. The sub-service of CreditPortal to be externalized.

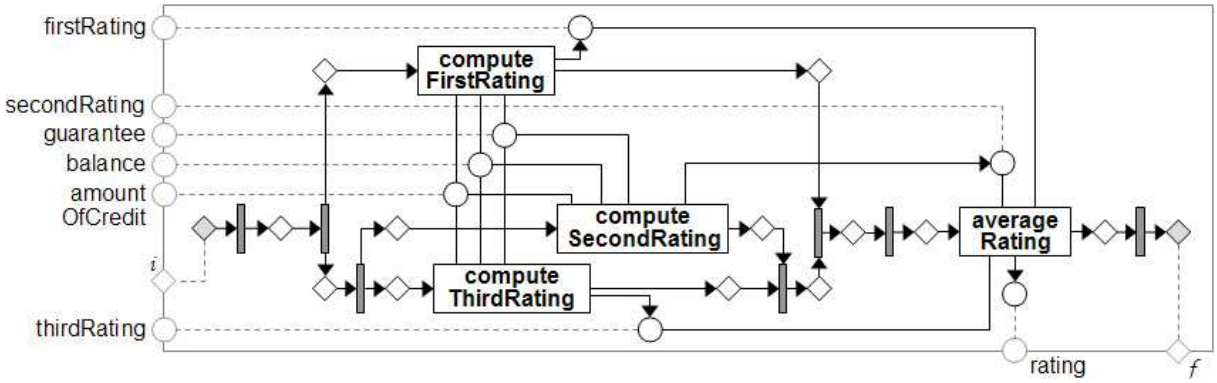

Figure 11. OCPR net representation of the RatingOne service.

As depicted in Figure 7, the CreditPortal process model is a sequence process whose left-most child is a repeat-until construct representing the customer authentication phase. The customer can repeatedly choose between logging in with an existing account (login) or creating a new account (createAccount) until either the log in to the system is successful (*validData* = true), or the system rejects the login definitively (*rejectedLogin* = true). Next, if the customer did not provide a valid login, CreditPortal terminates (invalidLogin). Otherwise, it continues by repeatedly asking the customer for the personal financial data (validateClientData) until either a valid information is uploaded (*validateResponse* = true) or the system rejects the credit request (*changeClientData* = false). Then, if the customer did not provide valid financial data, CreditPortal terminates (rejectClientData), otherwise the customer credit

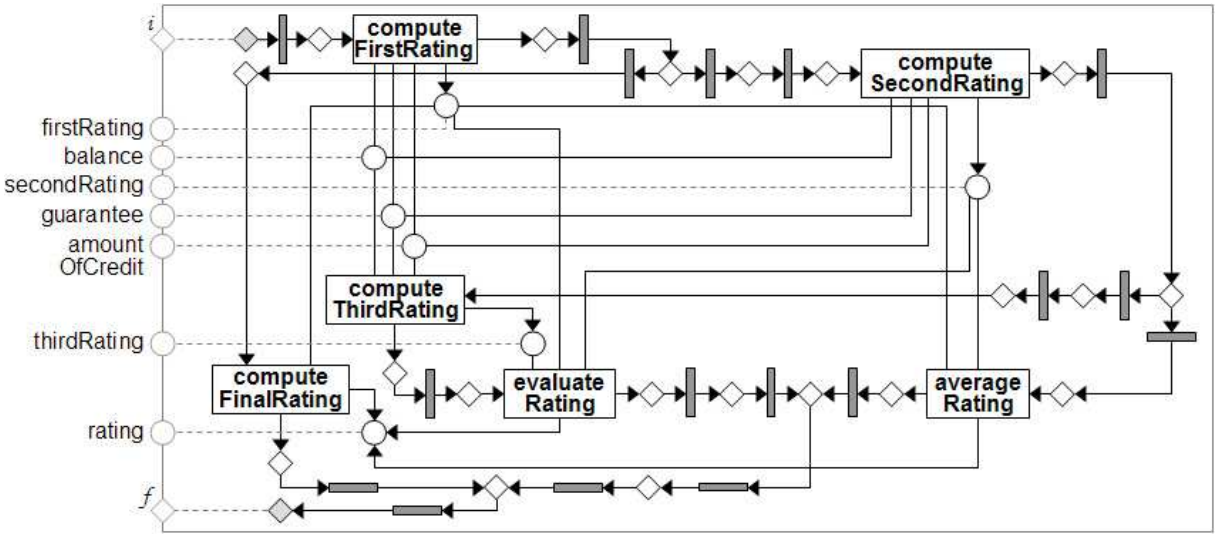

Figure 12. OCPR net representation of the RatingTwo service.

request evaluation phase starts. CreditPortal, taking into account the requested amount of credit, firstly evaluates the customer security (securityEvaluation), computes the customer rating (computingRating) and next decides whether or not to make an offer to the customer (makeOffer). If so (*makeOffer* = true), it builds the offer (buildOffer), formally confirms the offer (confirmOffer) and asks the customer for a final confirmation (userConfirmation). Next, if CreditPortal and the customer agree on the offer (*confirmation* = true and *userConfirmation* = true), the offer is finalized (finalizeCredit), otherwise CreditPortal rejects the credit request (rejectResponse). Instead, if CreditPortal does not want to make an offer (*makeOffer* = false), it can choose either to reject the credit request (rejectCredit) or to allow the customer to update the financial data (changeAmountOfCredit and changeGuarantee). In the latter case, the customer credit request evaluation phase is repeated.

Consider now the dotted section of the CreditPortal specification represented in Figure 7. It consists of a sequence of two atomic processes, i.e., securityEvaluation and computingRating, that takes as input the requested amount of credit, the balance and the provided guarantees of a customer, and then evaluates the reliability and the rating. Let us now suppose that the CreditPortal provider (i.e., the bank) wants to enhance its service and hence decides to externalize the CreditPortal section which computes customer rating, viz., the dotted area of Figure 7. For instance, suppose that two services which compute customer rating are available, that is, RatingOne and RatingTwo whose OWL-S process models (available at [1]) are illustrated in Figures 8 and 9, respectively. More precisely, RatingOne firstly computes three separate evaluations of the customer and then it returns an average rating, RatingTwo computes the customer rating and only if necessary (e.g., if the first rating exceeds a threshold value) it performs a second and possibly a third evaluation of the customer. RatingTwo may be more convenient for the bank, as it does not always compute three separate and expensive customer evaluations, yet, RatingOne provides a more accurate customer evaluation. In both cases, the bank needs to verify whether the dotted area of CreditPortal in Figure 7 can be replaced with RatingOne or RatingTwo, not affecting the internal behaviour of CreditPortal.

Such a replaceability problem can be solved by suitably exploiting our notion of behavioural congruence for Web services. We first translate the dotted area of CreditPortal as well as the services RatingOne and RatingTwo into OCPR nets, according to the OWL-S encoding presented in Section 5. The resulting nets are depicted in Figures 10, 11 and 12, respectively. Next, we check whether these three nets are equivalent with respect to Definition 4.7. If so, the dotted area of CreditPortal, RatingOne and RatingTwo can be used interchangeably. Yet, clearly, the nets of Figures 10, 11 and 12 are not equivalent, since they expose different interfaces, and they are obviously externally distinguishable. This is not surprising: the bank describes which are the information it needs for each customer (namely, balance and rating), while the additional information (i.e., firstRating, secondRating and thirdRating) provided by the enterprises may be used by the bank in order to choose a service according to some criteria.

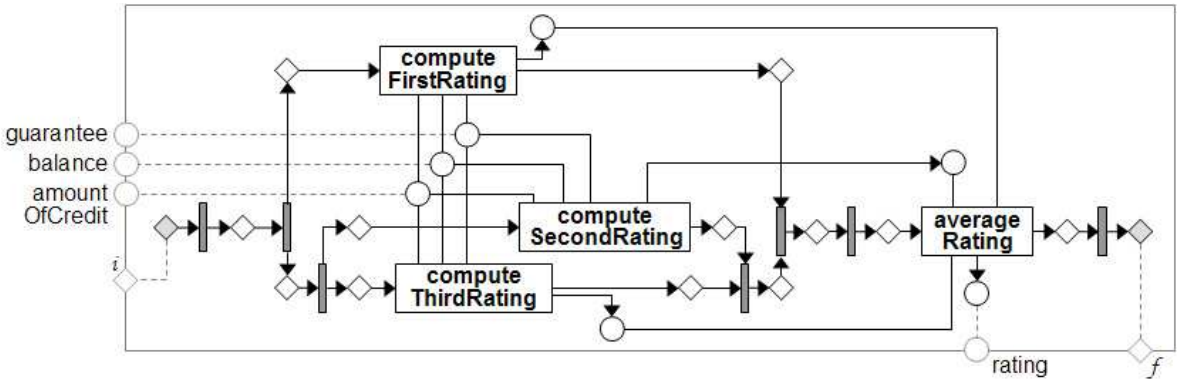

Figure 13. The net corresponding to ν_A[RatingOne] for $A = \{$firstRating, secondRating, thirdRating$\}$.

However, as one may notice in Figure 7, the bank is just interested in observing the final rating, not the intermediate ones. Hence, we can equate the nets of Figures 10, 11 and 12 after hiding firstRating, secondRating and thirdRating (e.g., Figure 13 illustrates the net resulting from applying such a hiding to the net of Figure 11). By doing so, these three nets become equivalent, i.e., they are structurally different, but externally indistinguishable. Hence, both RatingOne and RatingTwo can be employed to properly replace the sub-service of CreditPortal evaluating the customer reliability.

7. Related work

The successful introduction of observational equivalences for process calculi in the early 1980s spawned similar researches on nets, as witnessed for instance by the survey [37][5]. According to the taxonomy there, saturated bisimilarity is a state-based equivalence, since it encompasses a notion of interface (a set of observable places) and it is dictated by the way the firing relation crosses the interfaces. Indeed, our bisimilarity is reminiscent of *ST-equivalence*, as in [37, Def. 4.2.6].

This section does not aim at providing a survey of the field. First of all, the literature is very large, and its retelling would be definitely too space consuming, even if our main interest is in a *compositional* equivalence, thus restricting the area of possible intersection with former works. Additionally, our CPR

[5]The analysis there is restricted to *contact-free* C/E nets, i.e., such that the requirement $(M \cap t^\diamond) \subseteq {}^\diamond t$ of the enabling condition in Definition 2.4 holds for any reachable marking M.

nets have distinctive features, since data places act as sinks, thus any comparison should take that fact into account, putting an additional layer of complexity.

In the rest of the section we thus first focus on two issues that are closely related to the main novelties introduced in our framework: the use of the theory of reactive systems for obtaining a tractable equivalence, and the use of (equivalences on) nets for dealing with the specification of Web services and of their composition. We then finally overview a few works explicitly addressing service replaceability, even if not necessarily using net formalisms for modeling purposes.

7.1. Nets, open places and labels

Most recent formalisms for system specification come equipped with a reduction semantics: a suitable algebra of states is defined, and system evolution is represented by a relation between states. However, the lack of observable actions (either associated to the states, or to the reductions) inhibits the development of observational equivalences, which are often handier and more tractable.

Concerning Petri nets, the need of primitives for expressing the interaction with an environment was recognized early on, and notions of "net interface", intended as a subset of the items of the net, are already reported in [37]. Interfaces are key ingredients for defining an observation, as well as for expressing net operators: along this line, a classical approach is the Box Algebra [7]. The main difference with our proposal is in the use of a set of labels for obtaining a labelled reduction relation, on top of which to define the weak semantics.

Quite related to our solution are *open nets*: place/transitions nets where two distinguished sets of input and output places (where tokens may be added or removed, respectively) are identified, and then used to compose nets by coalescing places [6]. The approach in [6] however differs from ours in that the authors stick to P/T nets and introduce a dichotomy between input and output places, which reflects on the type of the net composition. We refer to [6], and the references therein, for a survey and comparison with other interface-based techniques, mostly important the *net components* proposed by Kindler [18].

The most important source of inspiration for our work is the theory of reactive systems [21], introducing a technique for synthesising labels with suitable minimality requirements from a reduction relation that guarantees that the bisimilarity on the derived labelled relation is a congruence. Indeed, our approach benefits from the general definition of interface deriving from the theory. Concerning Petri nets, the technique was first applied by Milner in [30], after implementing nets into a more complex graphical structure, bigraphs, and later by Sassone and Sobociński [41], whose labelled relation largely coincides with ours. The main difference with respect to [41], besides the use of our flavour of C/E nets, is the introduction of saturated bisimilarity and the corresponding characterisation by weak bisimilarity. Moreover, our equivalence is weak (the number of transitions is not observed) and interleaving (parallelism is reduced to non-determinism). To the best of our knowledge, the treatment of such a bisimilarity for nets, and its decidable characterisation, is original to our paper.

7.2. Nets for service equivalence

The application of Petri nets to the specification and the modelling of distributed systems has been around since their inception. Concerning Web services, the use of nets has been strongly advocated in the works by van der Aalst, see e.g. [43] and the position paper [3]. More specifically, he and his coauthors address

the issue of equivalences for nets in [5]: they propose a trace-based, probabilistic equivalence for nets which is however quite far from our proposal.

Tightly related to us is the work of Martens, which actually inspired some of our examples in Section 3. More precisely, we refer to [25], where van der Aalst's WF Petri nets (with disjoint sets of input, output and internal places) are analysed for checking structural properties of Web services. The author introduces there the notions of net module and net composition, which recall the open nets formalism mentioned above, and roughly coincide with our own solution.

Martens introduces in [25] the notion of net soundness and net usability, basically related to state reachability with respect to composition with suitable modules. We sketch here a solution for recasting those notions in terms of saturated bisimilarity. Let us start considering the OCPR net **1** depicted in Figure 14. Moreover, for any OCPR net $\mathcal{N}$, let $O = \langle i, f, OD \rangle$ its outer interface, and consider $\nu_{OD,O}$, i.e. the hiding operators that closes all the open data places of $\mathcal{N}$. According to [25], we say that a net $\mathcal{N}$ is *weakly sound* if and only if $\nu_{OD,O}[\mathcal{N}] \approx \mathbf{1}$; a context $C[-]$ *utilizes* a net $\mathcal{N}$ if $\nu_{OD,O}[C[\mathcal{N}]] \approx \mathbf{1}$; and a net $\mathcal{N}$ is *usable* if there exists a context $C[-]$ using it.

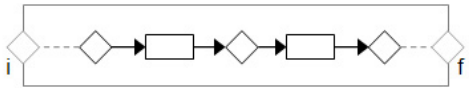

Figure 14. The OCPR net **1**.

Next, Martens considers a few notions of observational equivalence for his flavour of WF nets, discussing trace, bisimulation and simulation equivalences. He identifies the weakness of trace equivalence with respect to deadlock, as we echoed in Section 3. He further argues on bisimulation, reaching the conclusion that simulation is the most adequate equivalence. Note that Martens' notion of simulation (denoted here by $\equiv$) can be expressed by saturated bisimilarity. Indeed, $\mathcal{N} \equiv \mathcal{N}'$ if and only if $\nu_{OD,O}[C[\mathcal{N}]] \approx \mathbf{1} \Leftrightarrow \nu_{OD,O}[C[\mathcal{N}']] \approx \mathbf{1}$ for all possible contexts $C[-]$: intuitively, this means that the two nets are usable by exactly the same environments. Note that our solution is stricter, since $\approx \subseteq \equiv$, while the converse does not hold. Indeed, the nets M_3 and M_4 of Figure 4 in [26], now reproduced according to our notation in Figure 15, are equivalent according to Martens, even if they are not bisimilar.

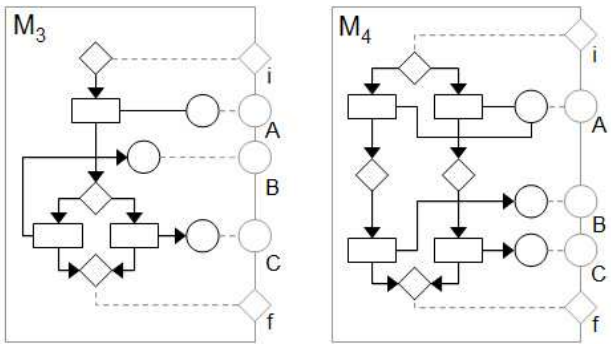

Figure 15. Two similar (according to Martens) OCPR nets that are not saturated bisimilar.

7.3. On formalisms for service replaceability

In the emerging world of Service-Oriented Computing – where applications are built by combining existing services – the issue of service replaceability gained a prominent role, and thus new approaches are often introduced. The discussion below briefly sums up some of the most recent proposals that we are aware of.

A logic-based approach for service replaceability has been recently presented in [36], where a context-specific definition of service equivalence is introduced. According to [36], given a μ-calculus formula ϕ describing some property, a service S, taking part in a specific context $C[-]$, can be replaced by a service T if ϕ holds also in $C[T]$. Intuitively, such a notion of context-specific replaceability relaxes the requirements imposed by a notion of service (bi)simulation like [8].

Another relaxed replaceability relation on services is induced by the definition of *interaction soundness* presented in [38]. Given an environment E, a service S in an orchestration $O[S]$ can be replaced by T if the interaction of $O[T]$ and E is *lazy sound*, that is, if the final node of the graph which represents the interaction of $O[T]$ and E can be reached from every initial node.

Although not presented in term of replaceability, the notion of *operating guidelines*, introduced in [27, 23] and employed in [22] to formally analyze the interactional behaviour of BPEL processes, also implicitly induces a replaceability relation on services — yet not compositional. An operating guideline is an automaton that concisely represents all the partners that properly interact with a service. A service S interacting with C can be replaced with a service T if T belongs to the operating guidelines of C.

A theory for checking the compatibility of service contracts based on a CCS-like calculus is presented in [15, 20]. Using a simple finite syntax (featuring the sequencing and external/internal choice constructors) to describe service contracts, they define a notion of preorder on processes (based on must testing) reflecting the ability of successfully interacting with clients. Such a notion induces a replaceability relation that, informally, allows one to replace a service S_1 with S_2 only if all clients compliant with S_1 are also compliant with S_2. Such a notion of replaceability is uncomparable with ours, as the former emerges from a synchronous model while the latter emerges from an asynchronous model. It is also worth noting that, in particular, [20] shows the existence of the *principal dual contract* (reminiscent of operating guideline), i.e., the smallest (namely, the most general) service contract that satisfies the client request.

Other interesting notions of service replaceability were introduced also in [12]. The approach in [12] models service behaviour as deterministic automata and defines substitutability with respect to three different notions of compatibility. In particular, *context dependent substitutability* states that given a service made of two sub-services S_1 and S_2, S_1 can be replaced by S_1' if S_1' is compatible with S_2, while *context independent substitutability* states that a service S can be replaced by a service S' if S' is compatible with all those services which are compatible with S (analogously to [24, 20]). Our notion of substitutability resembles the notion of context independent substitutability (w.r.t. definition of compatibility 1 of [12]) in an asynchronous and non-deterministic setting.

8. Concluding remarks

In this paper we first introduced *Open Consume-Produce-Read* (OCPR) nets to naturally model the behaviour of OWL-S Web services (although our approach can be extended to WS-BPEL services as well, in the line of [34]). Next, we defined the notion of *saturated bisimilarity* between two OCPR nets, which relies on the concepts of interface and CPR context. An interface is a set of open (i.e.,

externally observable) places which interact with the environment, while a CPR context represents a possible environment where a service can be embedded in.

The use of interfaces and contexts represents the main difference with respect to the standard theory of Petri nets. These two concepts are fundamental when modelling interactive systems, but they are not commonly exploited for net specification. Moreover, an important novelty of our approach is the fact that tokens in the data places of an OCPR net can be produced and read (analogously to contextual nets [31]), but never consumed. This choice allow us to model data persistency, i.e., once a data have been produced or received from another service, it remains available until the end of the execution of the service.

To the best of our knowledge, the proposed bisimilarity is the first equivalence employing Petri nets for Web services that features

- *weakness* – It abstracts from internal transition steps by equating structurally different, yet externally indistinguishable services. An obvious application of such a notion is for checking whether a (complex) service implements a given specification. Indeed, it is often the case that a service provider publishes simple service specifications by hiding unnecessary and/or confidential details of their implementations, as well as, similarly, a matchmaking system verifies whether a (composition of) service(s) matches a client query.

- *compositionality* – It is the largest bisimulation that is also a congruence. Thus, two equivalent services can be always used interchangeably. Consider, for instance, a complex application where a sub-service S fails (e.g., it becomes unavailable). S can be replaced with an equivalent service S' without changing the behaviour of the whole application.

- *decidability* – As the set of the states of an OCPR net is finite, the saturated bisimilarity is decidable. It can hence be employed by automated software in order to check whether two services are behaviourally equivalent.

We note that our relying on a standard notion of observational equivalence allows the reuse of existing theoretical techniques and practical tools, such as e.g. the characterizations of minimal equivalent nets and the the algorithms for calculating them. We are currently implementing the algorithm for building the transition system, and we are developing a more efficient algorithm for checking saturated bisimilarity based on *normalized minimization* [11].

Finally, it is important to note that – for the sake of simplicity – we used a single range of names for identifying the parameters of the presented services, so that the mapping among parameters of different services is obvious. Yet, it is often the case that different services employ different parameter names. In the case of OWL-S descriptions, each functional parameter is annotated with a concept defined in a shared ontology. Hence, we can (semi-)automatically determine the mapping between parameters of separate services by employing suitable tools for crossing ontologies. Otherwise, in the case of WS-BPEL [2], for example, such a mapping has to be provided manually. In this perspective our approach can be easily extended to WS-BPEL services, exploiting, e.g., a translation from BPEL processes to workflow nets in [34].

References

[1] http://www.di.unipi.it/∼corfini/owls/processmodels/.

[2] A. Alves et al.: WS-BPEL 2.0, 2006, `http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf`.

[3] van der Aalst, W.: Pi Calculus Versus Petri Nets: Let Us Eat "Humble Pie" Rather Than Further Inflate the "Pi Hype", *BPTrends*, **3**(5), 2005, 1–11.

[4] van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow Patterns, *Distributed and Parallel Databases*, **14**(1), 2003, 5–51.

[5] van der Aalst, W., Alves de Medeiros, A., Weijters, A.: Process Equivalence: Comparing Two Process Models Based on Observed Behavior, *Business Process Management*, LNCS *4102* (S. Dustdar, J. Fiadeiro, A. Sheth, Eds.), Springer, 2006.

[6] Baldan, P., Corradini, A., Ehrig, H., Heckel, R.: Compositional semantics for open Petri nets based on deterministic processes, *Mathematical Structures in Computer Science*, **15**(1), 2005, 1–35.

[7] Best, E., Devillers, R., Koutny, M.: The Box Algebra = Petri Nets + Process Expressions, *Information and Computation*, **178**(1), 2002, 44–100.

[8] Bonchi, F., Brogi, A., Corfini, S., Gadducci, F.: A Behavioural Congruence for Web services, *Fundamentals of Software Engineering*, LNCS *4767* (F. Arbab, M. Sarjani, Eds.), Springer, 2007.

[9] Bonchi, F., Brogi, A., Corfini, S., Gadducci, F.: Compositional Specification of Web Services via Behavioural Equivalence: A Case Study, *Applications and Theory of Petri Nets*, *LNCS 5062* (K. van Hee, R. Valk, Eds.), Springer, 2008.

[10] Bonchi, F., König, B., Montanari, U.: Saturated Semantics for Reactive Systems, *Logic in Computer Science*, IEEE Computer Society, 2006.

[11] Bonchi, F., Montanari, U.: Coalgebraic Models for Reactive Systems, *Concurrency Theory*, LNCS *4703* (L. Caires, V. Vasconcelos, Eds.), Springer, 2007.

[12] Bordeaux, L., Salaün, G., Berardi, D., Mecella, M.: When are Two Web Services Compatible?, *Technologies for E-Services*, LNCS *3324* (M.-C. Shan, U. Dayal, M. Hsu, Eds.), Springer, 2005.

[13] Brogi, A., Corfini, S.: Behaviour-aware discovery of Web service compositions, *International Journal of Web Services Research*, **4**(3), 2007, 1–25.

[14] Brogi, A., Corfini, S., Iardella, S.: From OWL-S descriptions to Petri nets, *ICSOC 2007 Workshops*, LNCS *4907* (E. D. Nitto, M. Ripeanu, Eds.), Springer, 2009.

[15] Castagna, G., Gesbert, N., Padovani, L.: A theory of contracts for Web services, *Principles of Programming Languages* (G. C. Necula, P. Wadler, Eds.), ACM 2008.

[16] Ehrig, H., König, B.: Deriving Bisimulation Congruences in the DPO Approach to Graph Rewriting, *Foundations of Software Science and Computation Structures*, LNCS *2987* (I. Walukiewicz, Ed.), Springer, 2004.

[17] Hamadi, R., Benatallah, B.: A Petri Net-based Model for Web Service Composition, *Australasian Database Conference, Conferences in Research and Practice in Information Technology 17* (K.-D. Schewe, X. Zhou, Eds.), Australian Computer Society, 2003.

[18] Kindler, E.: A Compositional Partial Order Semantics for Petri Net Components, *Applications and Theory of Petri Nets*, LNCS *1248* (P. Azèma, G. Balbo, Eds.), Springer, 1997.

[19] Lack, S., Sobociński, P.: Adhesive Categories, *Foundations of Software Science and Computation Structures*, LNCS *2987* (I. Walukiewicz, Ed.), Springer, 2004.

[20] Laneve, C., Padovani, L.: The Must Preorder Revisited, *Concurrency Theory*, LNCS *4703* (L. Caires, V. Vasconcelos, Eds.), Springer, 2007.

[21] Leifer, J., Milner, R.: Deriving bisimulation congruences for reactive systems, *Concurrency Theory*, LNCS *1877* (C. Palamidessi, Ed.), Springer, 2000.

[22] Lohmann, N., Massuthe, P., Stahl, C., Weinberg, D.: Analyzing Interacting BPEL Processes, *Business Process Management*, LNCS *4102* (S. Dustdar, J. L. Fiadeiro, A. Sheth, Eds.), Springer, 2006.

[23] Lohmann, N., Massuthe, P., Wolf, K.: Operating Guidelines for Finite-State Services, *Application and Theory of Petri Nets*, *LNCS 4546* (Y. Kleijn, A. Yakovlev, Eds.), Springer, 2007.

[24] Martens, A.: On Compatibility of Web Services, *Petri Net Newsletter*, **65**, 2003, 12–20.

[25] Martens, A.: Analyzing Web service based business processes, *Fundamental Approaches to Software Engineering*, LNCS *3442* (M. Cerioli, Ed.), Springer, 2005.

[26] Martens, A.: Consistency between Executable and Abstract Processes, *e-Technology, e-Commerce, and e-Services*, IEEE Computer Society, 2005.

[27] Massuthe, P., Reisig, W., Schmidt, K.: An Operating Guideline Approach to the SOA, *Annals of Mathematics, Computing & Teleinformatics*, **1**(3), 2005, 35–43.

[28] Mecella, M., Parisi-Presicce, F., Pernici, B.: Modeling *E*-service Orchestration through Petri Nets, *Technologies for E-Services*, LNCS 2444, (A. P. Buchmann, F. Casati, L. Fiege, M. Hsu, M.-C. Shan, Eds.), Springer, 2002.

[29] Milner, R.: A Calculus of Communicating Systems, LNCS *92*, Springer, 1980.

[30] Milner, R.: Bigraphs for Petri Nets, *Lectures on Concurrency and Petri Nets*, LNCS *3098* (J. Desel, W. Reisig, G. Rozenberg, Eds.), Springer, 2004.

[31] Montanari, U., Rossi, F.: Contextual Nets, *Acta Informatica*, **32**(6), 1995, 545–596.

[32] Narayanan, S., McIlraith, S.: Analysis and simulation of Web services, *Computer Networks*, **42**(5), 2003, 675–693.

[33] Newcomer, E.: *Understanding Web Services: XML, WSDL, SOAP, and UDDI*, Addison-Wesley, 2002.

[34] Ouyang, C., Verbeek, E., van der Aalst, W. M. P., Breutel, S., Dumas, M., ter Hofstede, A. H. M.: *Formal Semantics and Analysis of Control Flow in WS-BPEL*, Science of Computer Programming, 67(2-3), 2007, 162–198.

[35] OWL-S Coalition: OWL-S: Semantic Markup for Web Service, 2006, `http://www.ai.sri.com/daml/services/owl-s/1.2/overview/`.

[36] Pathak, J., Basu, S., Honavar, V.: On Context-Specific Substitutability of Web Services, *Web Services*, IEEE Computer Society, 2007.

[37] Pomello, L., Rozenberg, G., Simone, C.: A Survey of Equivalence Notions for net based Systems, *Advances in Petri nets*, LNCS *609* (G. Rozenberg, Ed.), Springer, 1992.

[38] Puhlmann, F., Weske, M.: Interaction Soundness for Service Orchestrations, *Service-oriented Computing*, LNCS *4294* (A. Dan, W. Winfried Lamersdorf, Eds.), Springer, 2006.

[39] Reisig, W.: *Petri Nets: An Introduction*, vol. 4 of *EATCS Monographs in Theoretical Computer Science*, Springer, 1985.

[40] Sassone, V., Sobociński, P.: Reactive systems over cospans, *Logic in Computer Science*, IEEE Computer Society, 2005.

[41] Sassone, V., Sobociński, S.: A Congruence for Petri Nets, *Petri Nets and Graph Transformation*, ENTCS *127* (H. Ehrig, J. Padberg, G. Rozenberg, Eds.), Elsevier, 2005.

[42] UDDI Coalition: The UDDI Technical White Paper, 2000, http://www.uddi.org/.

[43] Verbeek, H., van der Aalst, W.: Analyzing BPEL processes using Petri nets, *Applications of Petri Nets to Coordination, Workflow and Business Process Management* (D. Marinescu, Ed.), 2005.

[44] W3C: Simple Object Access Protocol (SOAP) 1.2, W3C working draft, 17 December 2001, 2001, http://www.w3.org/TR/2001/WD-soap12-part0-20011217/.

[45] W3C: WSDL 1.1, 2001, http://www.w3.org/TR/wsdl.

A. Appendix: Proof of Theorem 4.1

In this section a formal proof of Theorem 4.1 is given. Since the proof is quite long, we give a modular presentation of it, proving several intermediate steps. We first define some notational convention and then we enounce several lemmas that will be pivotal in the rest of the proof.

We say that a context $C[-]$ is *compatible* with an open net $\mathcal{N}$ if $C[\mathcal{N}]$ is well-defined, i.e., if the outer interface of $\mathcal{N}$ coincides with the inner interface of $C[-]$. Writing $C[\mathcal{N}]$ we implicitly mean that $C[-]$ is compatible with $\mathcal{N}$. Similarly, $Op(\mathcal{N})_f$ denotes the set $Op(\mathcal{N}) \setminus \{f\}$, implicitly assuming that f is the final place of the outer interface of the OCPR net $\mathcal{N}$.

The following lemma is tacitly used during the whole section.

Lemma A.1. (contexts preserves observations)
Let $(\mathcal{N}, M)$ and $(\mathcal{N}', M')$ be OCPR nets with markings, $C[-]$ a context compatible with the two nets, and $U \subseteq Op(\mathcal{N})_f (= Op(\mathcal{N}')_f)$ a marking. If $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$, then $Obs(C[\mathcal{N}], M \cup U) = Obs(C[\mathcal{N}'], M' \cup U)$.

Hence, when proving that a bisimulation is a congruence, we just consider the dynamic part, since the lemma above ensures that the static part (represented by the condition $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$) is preserved under context closure.

The next lemma just states that all the internal reductions of a component net are also enabled in the composite net or, in other words, that contextual composition can not inhibit internal reductions.

Lemma A.2. (contexts preserve reductions)
Let $(\mathcal{N}, M)$ be an OCPR net with marking, and $C[-]$ a compatible context. If $M \twoheadrightarrow_{\mathcal{N}} M_1$, then $M \twoheadrightarrow_{C[\mathcal{N}]} M_1$.

Weak bisimilarity is a congruence

In order to prove that $\approx_W = \approx_S$, we start here by proving that $\approx_W$ is a congruence, with respect to contexts and to possible markings over the contexts.

The first step is to prove that that $\approx_W$ is a congruence with respect to the addition of tokens to the open places of a net. For the sake of readability, in the following we denote $M \setminus \{f\}$ as $M - f$, implicitly assuming that f is the final place of the outer interface of the OCPR net at hand.[6]

Lemma A.3. ($\approx_W$ is a congruence with respect to markings)
Let $(\mathcal{N}, M)$ and $(\mathcal{N}', M')$ be OCPR nets with markings, such that $O_{\mathcal{N}} = O_{\mathcal{N}'}$, and $U \subseteq Op(\mathcal{N})_f$ a marking. If $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$, then $(\mathcal{N}, M \cup U) \approx_W (\mathcal{N}', M' \cup U)$ and $(\mathcal{N}, M - f) \approx_W (\mathcal{N}', M' - f)$.

Proof:
Suppose $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$ and let $o \in U$. The case $o \in M$ is obvious. If $o \notin M$, then $M \xrightarrow{+o}_{\mathcal{N}} M + o$; and since $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$, then $M' \xrightarrow{+o}_{\mathcal{N}'} M' + o$ and $(\mathcal{N}, M + o) \approx_W (\mathcal{N}', M' + o)$. By iterating the process, it holds $(\mathcal{N}, M + U) \approx_W (\mathcal{N}', M' + U)$.

Analogous reasoning holds for $M - f$. □

[6] Additionally, in the proofs $M \cup U$ and $M \cup \{o\}$ denote $M + U$ and $M + o$, respectively.

We now tackle the closure with respect to a compatible context.

Lemma A.4. ($\approx_W$ is a congruence with respect to contexts)
Let $(\mathcal{N}, M)$ and $(\mathcal{N}', M')$ be OCPR nets with markings, and $C[-]$ a context compatible with the two nets. If $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$, then $(C[\mathcal{N}], M) \approx_W (C[\mathcal{N}'], M')$.

Proof:
Let $\mathfrak{R}_1 = \{(C[\mathcal{N}], M + U), (C[\mathcal{N}'], M' + U) \mid (\mathcal{N}, M) \approx_W (\mathcal{N}', M')\}$ and $\mathfrak{R}_2 = \{(C[\mathcal{N}], M + U - f'), (C[\mathcal{N}'], M' + U - f') \mid (\mathcal{N}, M) \approx_W (\mathcal{N}', M')\}$ be two relations, for f' final place in $\mathcal{N}$ and for any $U \subseteq Op(C[\mathcal{N}])$ marking (hence possibly including places in $Op(\mathcal{N})$). We have to prove that $\mathfrak{R} = \mathfrak{R}_1 \cup \mathfrak{R}_2$ is a weak bisimulation.

Let us first consider the pairs in $\mathfrak{R}_1$.

Let $M + U \xrightarrow{+o}_{C[\mathcal{N}]} M + U + \{o\}$. The case $o \notin Op(\mathcal{N})$ is obvious. If $o \in Op(\mathcal{N})$, clearly $o \notin M$, hence $o \notin M'$, and also $M' + U \xrightarrow{+o}_{C[\mathcal{N}']} M' + U + \{o\}$. Since $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$, then $(\mathcal{N}, M + U + \{o\})\mathfrak{R}(\mathcal{N}', M' + U + \{o\})$.

Let $M + U \xrightarrow{-f}_{C[\mathcal{N}]} M + U - f$. The case $f \notin Op(\mathcal{N})$ is obvious. If $f \in Op(\mathcal{N})$, then f is the final place for $\mathcal{N}$ and $\mathcal{N}'$, too. Since $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$, then $(\mathcal{N}, M + U - f)\mathfrak{R}(\mathcal{N}', M' + U - f)$.

If $M + U \xrightarrow{\tau}_{C[\mathcal{N}]} M_1$, then there exists a transition $t \in T_{C[\mathcal{N}]}$ such that $(C[\mathcal{N}], M + U)[t\rangle(C[\mathcal{N}], M_1)$, with t belonging either to $\mathcal{N}$ or to $C[-]$.

- $t \in \mathcal{N}$. Then $M + U \xrightarrow{\tau}_{\mathcal{N}} M_1 + U$. By Lemma A.3, $(\mathcal{N}, M + U) \approx_W (\mathcal{N}', M' + U)$ and then $M' + U \xrightarrow{\tau}_{\mathcal{N}} M_1' + U$ with $(\mathcal{N}, M_1 + U) \approx_W (\mathcal{N}', M_1' + U)$. By Lemma A.2, $M' + U \xrightarrow{\tau}_{C[\mathcal{N}]} M_1' + U$ and we have that $(C[\mathcal{N}], M_1 + U)\mathfrak{R}(C[\mathcal{N}'], M_1' + U)$.

- $t \in C[-]$. Then, the only tokens of $\mathcal{N}$ used to perform t are those contained in the open places of $\mathcal{N}$. Since $(\mathcal{N}, M + U) \approx_W (\mathcal{N}', M' + U)$, then $Obs(\mathcal{N}, M + U) = Obs(\mathcal{N}', M' + U)$, and then the transition t can be performed also by $(C[\mathcal{N}'], M' + U)$. Moreover, the firing of t can produce new tokens in $Op(\mathcal{N})_{f'}$ and possibly consume the token in f'. All the other either consumed or produced tokens occur in the places of $C[-]$. Thus, the states reached by $(C[\mathcal{N}'], M' + U)$ and by $(C[\mathcal{N}], M + U)$ are again in the relation $\mathfrak{R}$.

For $\mathfrak{R}_2$ we can proceed as above. □

The lemmas above ensure that the proposition below holds.

Proposition A.1. $\approx_W$ is a congruence.

In other terms, $\approx_W$ is a congruence with respect to context closure and marking addition, that is, $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$ implies that $(C[\mathcal{N}], M \cup U - f) \approx_W (C[\mathcal{N}'], M' \cup U - f)$ for $f \in M$ final place of $\mathcal{N}$, and for any context $C[-]$ compatible with the two nets and marking $U \subseteq DP_{C[-]} \cup CP_{C[-]}$, i.e., possibly including closed places of $C[-]$.

Weak bisimilarity is saturated

We open the section by defining a notion of bisimulation that is intermediate with respect to $\approx_S$ and $\approx_W$. More precisely, it differs from saturated bisimulation because it is one-step, that is, only a single firing step is considered in the bisimulation game.

Definition A.1. (intermediate bisimulation)
A symmetric relation $\mathfrak{R} \subseteq \mathcal{MN} \times \mathcal{MN}$ is an *intermediate bisimulation* if whenever $(\mathcal{N}, M)\ \mathfrak{R}\ (\mathcal{N}', M')$ then

- $O_{\mathcal{N}} = O_{\mathcal{N}'}$ and $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$, and
- $\forall C[-].M[\rangle_{C[\mathcal{N}]}M_1$ implies $M' \twoheadrightarrow_{C[\mathcal{N}']} M_1'$ & $(C[\mathcal{N}], M_1)\ \mathfrak{R}\ (C[\mathcal{N}'], M_1')$.

The union of all intermediate bisimulations is called *intermediate bisimilarity* and denoted by $\approx_I$.

The proposition below is a standard result in the literature on bisimilarity.

Proposition A.2. $\approx_I = \approx_S$.

In order to prove Theorem 4.1, we need to introduce two special contexts.

Definition A.2. Let $\mathcal{N} = (N, O)$ be an OCPR net. Moreover, let $o \in Op(\mathcal{N})_f$ be an open place and o' a data place such that $o' \notin Op(\mathcal{N})$ Then, the contexts $ADD_o^O[-]$ and $SUB^O[-]$ are represented in Figure 16.

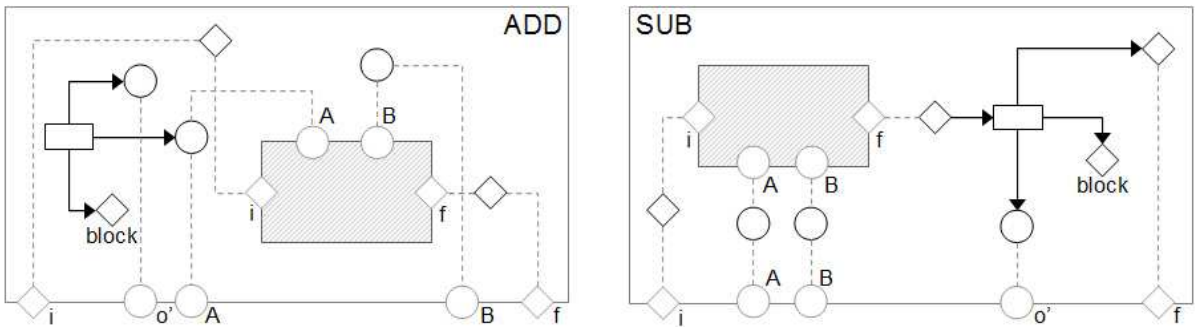

Figure 16. The context ADD_A^O and SUB^O, for interface $O = (i, f, \{A, B\})$.

The structure of the net is actually irrelevant, hence the superscript recording only the interface. Intuitively, $ADD_o^O[-]$ is a context taking a net with outer interface O and inserting a transition that adds a token to the place $o \in O$ and to another $o' \notin O$. The transition can be performed only once because of the control place "block" that inhibits further firings of the transition. SUB_f^O has an analogous behaviour, taking a net with outer interface O and inserting a transition that removes a token from the final place f and adds a token to the open data place o'. As for $ADD_o^O[-]$, the transition can be performed only once. The data place o' is used to look if the added transition have been performed. In fact, given an OCPR net N with outer interface O and a marking M, then $(ADD_o^O[\mathcal{N}], M)$ can reach a state where o' is observable if and only if it performs the new transition in ADD_o^O, because $o' \notin O$.

The behaviour of a net does not change, after inserting it into $ADD_o^O[-]$ and performing the added transition. This is stated by the following two lemmas.

Lemma A.5. Let $\mathcal{N} = (N, O)$ be an OCPR net, and let M be a marking of N. Then, $M \twoheadrightarrow_{ADD_o^O[\mathcal{N}]} M_1 \cup \{o, o', block\}$ iff $M \cup \{o\} \twoheadrightarrow_{\mathcal{N}} M_1 \cup \{o\}$.

Lemma A.6. Let $\mathcal{N} = (N, O)$ be an OCPR net, and let M be a marking of N. If $(ADD_o^O[\mathcal{N}], M \cup \{o, o', block\}) \approx_S (ADD_o^O[\mathcal{N}'], M' \cup \{o, o', block\})$, then $(\mathcal{N}, M \cup \{o\}) \approx_S (\mathcal{N}', M' \cup \{o\})$.

Analogous lemmas hold for $SUB_f^O[-]$. There results are used to prove that $\approx_S$ is a congruence also with respect to markings.

Lemma A.7. ($\approx_S$ is a congruence with respect to markings)
Let $(\mathcal{N}, M)$, $(\mathcal{N}', M')$ be OCPR nets with markings, such that $O_{\mathcal{N}} = O_{\mathcal{N}'}$, and $U \subseteq Op(\mathcal{N})_f$ a marking. If $(\mathcal{N}, M) \approx_S (\mathcal{N}', M')$, then $(\mathcal{N}, M \cup U) \approx_S (\mathcal{N}', M' \cup U)$ and $(\mathcal{N}, M - f) \approx_S (\mathcal{N}', M' - f)$.

Proof:
Let $o \in U$ an open place, and let us assume that $o \notin M$. Consider the relation $\mathfrak{R} = \{((\mathcal{N}, M + o), (\mathcal{N}', M' + o)) \mid (\mathcal{N}, M) \approx_S (\mathcal{N}', M')\}$: We prove that $\mathfrak{R}$ is a saturated bisimulation, and then we iterate for all places in U.

Suppose that $M + o \twoheadrightarrow_{C[\mathcal{N}]} M_1 + o$, then, by Lemma A.5, we obtain that $M \twoheadrightarrow_{ADD_o^O[C[\mathcal{N}]]} M_1 + o + o' + block$. Since $(\mathcal{N}, M) \approx_S (\mathcal{N}', M')$, then $M' \twoheadrightarrow_{ADD_o^O[C[\mathcal{N}']]} M_1' + o + o' + block$ and $(ADD_o^O[C[\mathcal{N}]], M_1 + o + o' + block) \approx_S (ADD_o^O[C[\mathcal{N}']], M_1' + o + o' + block)$. By Lemma A.5, $M' \twoheadrightarrow_{C[\mathcal{N}']} M_1' + o$, and by Lemma A.6 we have $(C[\mathcal{N}], M_1) \approx_S (C[\mathcal{N}'], M_1')$.

Analogously for $-f$. □

Theorem A.1. $\approx_S = \approx_W$

Proof:
In order to prove $\approx_W \subseteq \approx_S$, we have to prove that $\approx_W$ is an intermediate bisimulation. Suppose that $(\mathcal{N}, M) \approx_W (\mathcal{N}', M')$ and $M \xrightarrow{\tau}_{C[\mathcal{N}]} M_1$. By Lemma A.4 we have $(C[\mathcal{N}], M) \approx_W (C[\mathcal{N}'], M')$, hence $M' \twoheadrightarrow_{C[\mathcal{N}']} M_1'$ with $(C[\mathcal{N}], M_1) \approx_S (C[\mathcal{N}'], M_1')$.

In order to prove $\approx_S \subseteq \approx_W$, we have to prove that $\approx_S$ is a weak bisimulation. Suppose that $(\mathcal{N}, M) \approx_S (\mathcal{N}', M')$.

If $M \xrightarrow{+o}_{\mathcal{N}} M_1 + o$, then o is an open place with $o \notin M$ and thus $o \notin M'$. Then $M' \xrightarrow{+o}_{\mathcal{N}'} M_1' + o$ and by Lemma A.7 we have $M_1 + o \approx_S M_1' + o$.

If $M \xrightarrow{-f}_{\mathcal{N}} M_1 - f$, we can apply an analogous reasoning.

If $M \xrightarrow{\tau}_{\mathcal{N}} M_1$, then trivially $M \twoheadrightarrow_{\mathcal{N}} M_1$, hence $M' \twoheadrightarrow_{\mathcal{N}'} M_1'$ with $(\mathcal{N}, M_1) \approx_S (\mathcal{N}', M_1')$. □

B. Appendix: On semi-saturated bisimilarity

We already remarked in Section 4.3 that the theory of reactive systems ensures that, for a suitable choice of labels, the (strong) bisimilarity on a derived LTS is a congruence [21]. That distilled bisimilarity, however, usually does not coincide with the saturated bisimulation on the original, unlabelled TS. In [10] the first author, together with König and Montanari, discusses the problem and introduces the notion of *semi-saturated* bisimilarity that coincides with the saturated one. In the version of the present paper for the conference proceedings [8] we exploited those results, presenting the definition below.

Definition B.1. (semi-saturated bisimulation)
A symmetric relation $\mathfrak{R} \subseteq \mathcal{MN} \times \mathcal{MN}$ is a *semi saturated bisimulation* if whenever $(\mathcal{N}, M)\ \mathfrak{R}\ (\mathcal{N}', M')$ then

- $O_{\mathcal{N}} = O_{\mathcal{N}'}$ and $Obs(\mathcal{N}, M) = Obs(\mathcal{N}', M')$,
- $M \stackrel{+o}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \cup \{o\} \twoheadrightarrow_{\mathcal{N}'} M_1'$ and $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$,
- $M \stackrel{-f}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \setminus \{f\} \twoheadrightarrow_{\mathcal{N}'} M_1'$ and $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$, and
- $M \stackrel{\tau}{\rightarrow}_{\mathcal{N}} M_1$ implies $M' \twoheadrightarrow_{\mathcal{N}'} M_1'$ and $(\mathcal{N}, M_1)\ \mathfrak{R}\ (\mathcal{N}', M_1')$.

The union of all semi-saturated bisimulations is called *semi-saturated bisimilarity* ($\approx_{SS}$).

The key theorem of [8] is now simply stated below.

Theorem B.1. $\approx_S = \approx_{SS}$.

Theorem 4.1 of the present paper supersedes the result above, since it uses the standard notion of weak bisimilarity to recast the saturated one.